BAYESAME: BAYESIAN ACTIVE MODEL EVALUATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Evaluating large generative models across benchmarks is time-consuming and costly, motivating methods that can estimate full benchmark performance by evaluating models on only a subset of items (*coreset*). While current literature mostly requires the practitioner to input a coreset size, we introduce BAYESAME, a sequential active Bayesian framework that uses historical data and an information-gain criterion to dynamically select items to include in the coreset thus allowing automatic determination of its size. Additionally, we show how a multi-target extension further reduces coreset size by leveraging model correlations. Extensive experiments demonstrate that BAYESAME consistently outperforms existing methods. Crucially, our analysis resolves recent skepticism in the literature by showing that a) non-random coreset selection is advantageous over random selection and b) one can reliably improve upon the random sample mean baseline.

1 INTRODUCTION

The scale and frequency of model evaluations make benchmarking large generative models time-consuming and computationally expensive (Liang et al., 2023). To mitigate this, a growing number of methods estimate full benchmark performance by evaluating models on only a subset of items, known as a *coreset*. Such methods typically require the practitioner to input a coreset size (Berrada et al., 2025; Bowyer et al., 2026; Fisch et al., 2026; Huang et al., 2026; Kipnis et al., 2025; Kossen et al., 2021; Liao et al., 2025; Liu et al., 2026; Perlitz et al., 2024; Saranathan et al., 2025; Vivek et al., 2024; Zhang et al., 2025). While this is most suitable when facing strict computational limits, in scenarios where reliable performance estimation takes priority over efficiency, an evaluation method should also be capable of automatically determining a coreset size that reflects this priority. In addition, the predominant approach of leveraging evaluations from historical models (*reference models*) has largely operated within the regime in which the models being evaluated (*target models*) fall within the performance range of these reference models (*interpolation regime*). Zhang et al. (2025) underscored the importance of also testing efficient model evaluation methods in regimes where the target models significantly deviate from this range (*extrapolation regimes*) to obtain a more accurate assessment of a method's robustness. In particular, through a study that considered both regimes, the authors questioned the utility of non-random coreset selection and highlighted the difficulty of outperforming the mean of a random coreset as a performance estimate in extrapolation.

In this work, we focus on efficient model evaluation leveraging reference models, specifically targeting automatic determination of the coreset size. We propose BAYESAME, a sequential Bayesian method that models performance as a random variable via underlying abilities whose joint prior distribution encodes the belief that the target model behaves similarly to the reference models. The posterior distribution over these abilities is used to derive performance estimators, quantify performance uncertainty, and select items to add to the coreset via an information-gain criterion. The coreset is iteratively augmented until the performance estimate fluctuation and the performance uncertainty fall below their respective user-defined thresholds. In the setting where multiple target models are being evaluated, we capture correlation among them using a linear model of coregionalization (Álvarez et al., 2012; Journel & Huijbregts, 1978) with the goal of further reducing the coreset size. To validate our approach, we comprehensively evaluate BAYESAME against sequential adaptations of existing methods.

Our main contributions are summarized as follows:

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

- We formulate efficient model evaluation with automatic coreset-size determination as a Bayesian sequential process that augments the coreset until the performance estimate fluctuation and the performance uncertainty fall below their respective user-defined thresholds.
- We develop BAYESAME, a computationally efficient framework with robust performance estimators, information-gain-based active item selection, and a multi-target extension that exploits correlations across target models to further reduce coreset size.
- We conduct an extensive empirical analysis across varying numbers of reference models, interpolation/extrapolation regimes, and different degrees of target model correlation, showing that BAYESAME outperforms sequential adaptations of existing methods for automatic coreset-size determination, while also remaining competitive when the coreset size is predetermined. Our results challenge recent claims: non-random coreset selection improves performance, and several methods reliably outperform the random sample mean baseline.
- We demonstrate that leveraging available response log-likelihoods, rather than using binary scores, greatly improves performance estimation.

2 PROBLEM SETUP

Consider a benchmark $E = \{(x_i, y_i)\}_{i=1}^N$, where x_i is the i-th input instance and y_i its ground-truth response. Let $\mathcal{T} = \{t_1, \dots, t_{K_t}\}$ be a set of K_t *target models*, and let s_i^t denote the *score* of target model $t \in \mathcal{T}$ on item (x_i, y_i). Our goal is to estimate the full benchmark performance $R_t^\star = \frac{1}{N}\sum_{i=1}^N s_i^t$ by evaluating each target model t only on a subset of items $\mathcal{C}^t \subseteq E$, which we refer to as a *coreset* (throughout, we use E and $\mathcal{C}^t$ to indicate the indices of the benchmark items rather than the items themselves). We focus on the setting where the full benchmark scores for a set of K_r *reference models* $\{r_1, \dots, r_{K_r}\}$ are available, and scores are binary or bounded continuous (e.g., $s_i^t \in [0, 1]$ represents the probability assigned to the ground-truth response), although the framework is applicable for $s_i^t \in \mathbb{R}$. We consider settings where users prioritize reliable performance estimation over minimizing cost, but do not know the coreset size required to achieve this, and therefore focus on methods that support both user-specified and automatically determined coreset sizes.

3 BAYESAME

BAYESAME is a sequential Bayesian method that defines the full benchmark performance as $R_t = \frac{1}{N}\sum_{i=1}^N S_i^t$, where S_i^t is a random variable representing the score on benchmark item (x_i, y_i). Each S_i^t is assumed to be a noisy version of a latent *ability*, shared across items with the same reference model scores. The abilities are assigned a joint Gaussian prior distribution to encode the belief that, if the reference models achieved similar scores on two groups of items, the target model likely exhibits similar behavior. At each iteration, the posterior of the latent abilities is used to derive performance estimators, quantify performance uncertainty, and select an item to add to the coreset via an information gain criterion. The sequential process terminates when the performance fluctuation and uncertainty fall below user-defined thresholds.

3.1 SINGLE-TARGET SETTING

We first consider a single target model, omitting its index throughout this section. Let $(s_i^{r_1}, \dots, s_i^{r_{K_r}})$ be the vector of scores achieved by the K_r reference models on item (x_i, y_i). Suppose that the reference models yield B unique score vectors across the N benchmark items, which we denote as the set $\{\phi_1, \dots, \phi_B\}$, partitioning the items into B disjoint buckets. We consider a random vector $\theta = (\theta_1, \dots, \theta_B)$, where θ_b represents the target model's underlying ability on items belonging to bucket b, and assign it a Gaussian prior distribution $p(\theta) = \mathcal{N}(\mu^\theta, \Sigma^\theta)$ with

$$\mu_b^\theta = \tfrac{1}{K_r}\phi_b^\top \mathbf{1}_{K_r}, \qquad \Sigma_{b,b'}^\theta = \alpha \exp\left(-\tfrac{\beta}{K_r}\|\phi_b - \phi_{b'}\|^2\right), \quad \alpha, \beta > 0, \tag{1}$$

where $\mathbf{1}_{K_r}$ denotes a K_r-dimensional column vector of ones, α and β are parameters to be learned, and $\|\phi_b - \phi_{b'}\|^2$ is the squared Euclidean distance. This form of the covariance encodes the belief that buckets with similar reference scores yield highly correlated latent abilities for the target model.

We model the target model's scores as a random vector $S = (S_1, \dots, S_N)$ whose components are conditionally independent given θ. Specifically, if (x_i, y_i) belongs to bucket b, $p(S_i \mid \theta) =$

$\mathcal{N}(\theta_b, N_b\sigma^2)$, where N_b is the bucket size. Let H denote the $N \times B$ bucket indicator matrix. Then $p(S \mid \theta) = \mathcal{N}(H\theta, D)$, where D is diagonal with $D_{i,i} = N_b\sigma^2$ for items in bucket b. This noise model tempers the likelihood contribution of larger buckets. By scaling the individual item variance by N_b, we ensure that the variance of the average bucket score remains constant at σ^2, regardless of how many items the bucket contains: $\mathrm{Var}(\frac{1}{N_b}\sum_{i\in\text{bucket } b} S_i|\theta) = \frac{1}{N_b^2}\sum_{i\in\text{bucket } b}\mathrm{Var}(S_i|\theta) = \frac{1}{N_b^2}(N_b \cdot N_b\sigma^2) = \sigma^2$.

This Gaussian model[1] admits closed-form posterior inference. Let $s_\mathcal{C}$ denote the scores observed from evaluating the items in $\mathcal{C}$. We have $p(\theta \mid s_\mathcal{C}) = \mathcal{N}(\mu^\theta_{|\mathcal{C}}, \Sigma^\theta_{|\mathcal{C}})$, which gives:

$$\mathbb{E}[R \mid s_\mathcal{C}] = \tfrac{1}{N}\left(\textstyle\sum_{i\in\mathcal{C}} s_i + \sum_{b=1}^{B}(N_b - n_b)\mu^\theta_{b|\mathcal{C}}\right),$$
$$\mathrm{Var}(R \mid s_\mathcal{C}) = \tfrac{1}{N^2}\textstyle\sum_{b=1}^{B}(N_b - n_b)\left(\sigma^2 N_b + \sum_{b'=1}^{B}(N_{b'} - n_{b'})\Sigma^\theta_{b,b'|\mathcal{C}}\right), \tag{2}$$

where n_b is the number of evaluated items in bucket b. Upon full evaluation, $n_b = N_b$ for all b, so the posterior mean equals $R^\star$ and the variance vanishes. The mean $\mu^\theta_{|\mathcal{C}}$ and covariance $\Sigma^\theta_{|\mathcal{C}}$ can be efficiently computed in $\mathcal{O}(\min(|\mathcal{C}|, B)^2)$ per iteration between hyperparameter refits (see App. A.1).

Performance Estimator We construct an estimator $\hat{R}$ of $R^\star$ by adapting our approach to the number of unique reference score vectors in the benchmark, introducing a small margin $N_0 > 0$ to allow for slight deviations from uniqueness. In the *nearly-unique reference regime* $B \geq N - N_0$, the Bayes estimator under squared-error loss for the specified model is the posterior expectation:

$$\hat{R}_{\text{Bayes}} = \mathbb{E}[R \mid s_\mathcal{C}] = \tfrac{1}{N}\left(\textstyle\sum_{i\in\mathcal{C}} s_i + \sum_{b=1}^{B}(N_b - n_b)\mu^\theta_{b|\mathcal{C}}\right). \tag{3}$$

However, using Eq. (3) in the *non-unique reference regime* $B < N - N_0$ presents a potential issue: if a heavily populated bucket b has a large number of unevaluated items ($n_b \ll N_b$), its posterior mean $\mu^\theta_{b|\mathcal{C}}$ exerts a high weight in the overall estimate, amplifying any estimation errors. In this regime, we instead use the following estimator:

$$\hat{R}_{\text{CV}} = \tfrac{1}{n}\textstyle\sum_{i\in\mathcal{C}} s_i + \sum_{b=1}^{B}\left(\frac{N_b}{N} - \frac{n_b}{n}\right)\mathbb{E}[\theta_b \mid s_\mathcal{C}] = \tfrac{1}{n}\sum_{i\in\mathcal{C}} s_i + \sum_{b=1}^{B}\left(\frac{N_b}{N} - \frac{n_b}{n}\right)\mu^\theta_{b|\mathcal{C}}, \tag{4}$$

where n indicates the size of the coreset $\mathcal{C}$. This estimator serves as an in-sample proxy for the unbiased control variate estimator given in App. A.2. We can unify both regimes using:

$$\hat{R} = w\textstyle\sum_{i\in\mathcal{C}} s_i + \sum_{b=1}^{B}\left(\frac{N_b}{N} - n_b w\right)\mu^\theta_{b|\mathcal{C}}, \tag{5}$$

where $w = \frac{1}{n}$ if $B < N - N_0$ and $w = \frac{1}{N}$ otherwise.

Selection Strategy At each iteration of the sequential process, we use an information-gain criterion to select the next item on which to evaluate the target model. Specifically, we choose an item that maximally reduces the expected uncertainty of the model's performance R, measured via differential entropy: $\alpha_{\text{IG}}(i) = \mathcal{H}(R \mid s_\mathcal{C}) - \mathbb{E}_{S_i|s_\mathcal{C}}[\mathcal{H}(R \mid s_\mathcal{C}, S_i)]$. Assuming candidate item i belongs to bucket b, due to the Gaussian nature of the posterior, this expected information gain simplifies to:

$$\alpha_{\text{IG}}(i) = \mathbb{E}_{S_i|s_\mathcal{C}}\left[\tfrac{1}{2}\log\left(\frac{\mathrm{Var}(R \mid s_\mathcal{C})}{\mathrm{Var}(R \mid s_\mathcal{C}, S_i)}\right)\right] = -\tfrac{1}{2}\log\left(1 - \frac{\left(\sum_{b'=1}^{B}(N_{b'} - n_{b'})\Sigma^\theta_{b,b'|\mathcal{C}} + \sigma^2 N_b\right)^2}{N^2\,\mathrm{Var}(R \mid s_\mathcal{C})(\Sigma^\theta_{b,b|\mathcal{C}} + \sigma^2 N_b)}\right). \tag{6}$$

Note that all items in the same bucket have the same expected information-gain value. In the non-unique reference regime with imbalanced bucket sizes, purely active selection can degrade performance. This occurs because the target model may exhibit fine-grained variations in behavior that are not captured by the coarse grouping of the reference models. Consequently, when $B < N - N_0$, we default to random selection, which ensures that the expected number of sampled items from each bucket remains proportional to its size. For continuous scores, reference vectors may be nearly identical without being exactly equal so we compute an approximate bucket count based on an ℓ_∞ tolerance (see App. C.2) and default to random selection when this count falls below $N - N_0$. We also explored alternative selection strategies that operate on batches of items rather than individual items, including non-myopic approaches. However, these provided similar performance (see App. B.1).

[1] A more principled approach to modeling binary and bounded continuous scores would explicitly model their support. Exploring this direction revealed that the approximations required not only introduced substantial computational overhead but also resulted in inferior performance (see Appendix B.3).

Stopping Criterion A key objective is to determine the coreset size automatically, rather than requiring it as input from the user. We therefore monitor the uncertainty of R as items are evaluated, using the width of the 95% credible interval of R: $q_{97.5}(R \mid s_{\mathcal{C}}) - q_{2.5}(R \mid s_{\mathcal{C}}) = 2z_{0.975}\sqrt{\mathrm{Var}(R \mid s_{\mathcal{C}})}$, where $z_{0.975}$ denotes the standard normal z-score and $\mathrm{Var}(R \mid s_{\mathcal{C}})$ is given in Eq. (2). We sequentially augment the coreset until a two-condition stopping criterion is met: (i) the range of the performance estimate $\hat{R}$ over the last W iterations is at most ϵ_1; and (ii) $2z_{0.975}\sqrt{\mathrm{Var}(R \mid s_{\mathcal{C}})} \leq \epsilon_2$. The threshold ϵ_2 represents the maximum acceptable width of this uncertainty interval. For additional discussion, see App. A.3.

Algorithm We summarize the method in Alg. 1. Starting with an empty coreset $\mathcal{C}$, BAYESAME iteratively selects the benchmark item maximizing the expected information gain $\alpha_{\mathrm{IG}}(i)$ when $B \geq N - N_0$ (or at random otherwise). The target model is evaluated on this item and the posterior mean $\mu^{\theta}_{|\mathcal{C}}$ and covariance $\Sigma^{\theta}_{|\mathcal{C}}$ are updated, along with the performance estimate $\hat{R}$. Every F iterations, the covariance hyperparameters α and β are optimized by minimizing the negative log-marginal likelihood $\mathcal{L}(\alpha, \beta)$ of the coreset scores (see App. A.4). This sequential process terminates once the two-condition stopping criterion is satisfied.

Algorithm 1 BAYESAME SINGLE-TARGET

Require: Benchmark E; noise variance σ^2; margin N_0; stopping thresholds ϵ_1, ϵ_2 and window-length W; OPTIMIZER, hyperparameter initial values α_0, β_0 and update frequency F
Initialize coreset $\mathcal{C} \leftarrow \emptyset$
Initialize hyperparameters $\alpha \leftarrow \alpha_0, \beta \leftarrow \beta_0$
for $j \in \{1, \ldots, N\}$ **do**
  if $B \geq N - N_0$ **then**
    Select item via expected information gain:
    $i^* = \arg\max_{i \in E \setminus \mathcal{C}} \alpha_{\mathrm{IG}}(i)$ ▷ Eq. (6)
  else
    Select item via random sampling: $i^* \sim \mathrm{Uniform}(E \setminus \mathcal{C})$
  end if
  Obtain score s_{i^*} and add i^* to the coreset: $\mathcal{C} \leftarrow \mathcal{C} \cup \{i^*\}$
  if $j \bmod F = 0$ **then**
    $\alpha, \beta \leftarrow \mathrm{OPTIMIZER}(\mathcal{L}(\alpha, \beta), \alpha, \beta)$ ▷ Eq. (10)
  end if
  Compute $\mu^{\theta}_{|\mathcal{C}}$ and $\Sigma^{\theta}_{|\mathcal{C}}$ ▷ Eq. (7) or Eq. (8)
  Compute performance estimate $\hat{R}$ ▷ Eq. (5)
  Set $\hat{R}_j \leftarrow \hat{R}$
  if $j \geq W$ **and** $\max_{l \in [j-W+1, j]}(\hat{R}_l) - \min_{l \in [j-W+1, j]}(\hat{R}_l) \leq \epsilon_1$ **and** $2z_{0.975}\sqrt{\mathrm{Var}(R \mid s_{\mathcal{C}})} \leq \epsilon_2$ **then**
    break
  end if
end for
Return $\hat{R}$

3.2 MULTI-TARGET SETTING

For multiple target models $\mathcal{T} = \{t_1, \ldots, t_{K_t}\}$, we could apply the single-target method of Section 3.1 independently. However, jointly modeling target models that exhibit correlated performance can improve estimation accuracy and reduce the required coreset size. We therefore use a linear model of coregionalization (Álvarez et al., 2012; Journel & Huijbregts, 1978).

We capture cross-model correlations by assuming that their abilities are driven by L shared, independent latent random vectors $u_l \in \mathbb{R}^B$. Specifically, let θ^t_b represent the latent ability of target model t on bucket b, and let $\theta^t = (\theta^t_1, \ldots, \theta^t_B)$. We model θ^t as:

$$\theta^t = \mu^{\theta^t} + \textstyle\sum_{l=1}^{L} w^t_l u_l,$$

where μ^{θ^t} is defined as in Eq. (1), $w^t_1, \ldots, w^t_L$ are model-specific weights to be learned, and $u_l \sim \mathcal{N}(0, \Sigma^{\theta,l})$, where $\Sigma^{\theta,l}$ is a covariance matrix specific to latent vector l. This induces a Gaussian prior distribution over $\theta = (\theta^{t_1}, \ldots, \theta^{t_{K_t}})$ given by:

$$p(\theta) = \mathcal{N}\left(\mu^{\theta}, \textstyle\sum_{l=1}^{L} w_l w_l^{\top} \otimes \Sigma^{\theta,l}\right),$$

where $\otimes$ denotes the Kronecker product, $w_l = (w^{t_1}_l, \ldots, w^{t_{K_t}}_l)^{\top}$ and $\mu^{\theta} = (\mu^{\theta^{t_1}}, \ldots, \mu^{\theta^{t_{K_t}}})$. This formulation provides a flexible yet computationally efficient way to capture correlation across both buckets and models by combining bucket-specific ($\Sigma^{\theta,l}_{b,b'}$) and model-specific (w^t_l) terms. To provide greater modeling flexibility, we use two covariance families with complementary inductive biases: the squared exponential covariance defined as in Eq. (1), which imposes strict smoothness, and a Matérn-3/2 kernel which imposes weaker smoothness. Because the degree of correlation between the target models is typically unknown *a priori*, we set $L = K_t$. This enables the framework to dynamically adapt: it captures strong correlations by sharing latent components across models, yet

it retains the ability to recover the fully independent case. The scores are then modeled similarly to the single-target setting described in Section 3.1.

Selection Strategy In contrast to the single-target setting, the multi-target selection step requires simultaneously selecting an item and a target model to evaluate on that item, substantially expanding the search space. Furthermore, relying on a purely greedy information-gain criterion across this joint item-model space can bias performance estimates, particularly during early stages when correlations across target models are still being learned. To mitigate this bias and establish a robust initial estimate of the correlation structure, we depart from the empty coreset initialization used in the single-target setting. Instead, we randomly select 10% of the benchmark items and evaluate all target models on them. After this initialization step, we deploy the following selection strategy, which maintains some degree of exploration.

We iterate deterministically through the target models. For a given model t, we select the item that maximizes the expected information gain: $\alpha^t_{\text{IG}}(i) = \mathcal{H}(R^t \mid s_{\mathcal{C}}) - \mathbb{E}_{S^t_i}[\mathcal{H}(R^t \mid s_{\mathcal{C}}, S^t_i)]$, where $s_{\mathcal{C}} = \{s^{t_1}_{\mathcal{C}^{t_1}}, \ldots, s^{t_{K_t}}_{\mathcal{C}^{t_{K_t}}}\}$. This approach substantially reduces the search space, maintaining a selection complexity identical to that of the single-target setting. We iteratively augment the coresets $\{\mathcal{C}^{t_1}, \ldots, \mathcal{C}^{t_{K_t}}\}$ until all target models have met the two-condition stopping criterion detailed above. We also investigated an alternative average marginalized information-gain strategy, see App. C.5.

4 Related work

The issue of computational inefficiency in model evaluation is primarily tackled from two, sometimes intersecting, directions: constructing smaller, static versions of existing benchmarks (or developing methods for reducing benchmarks) (Bean et al., 2025; Kipnis et al., 2025; Zhao et al., 2025), and estimating performance without fully evaluating the target model (Berrada et al., 2025; Bowyer et al., 2026; Fisch et al., 2026; Huang et al., 2026; Kipnis et al., 2025; Kossen et al., 2021; Hsu & Shekhar, 2026; Liao et al., 2025; Liu et al., 2026; Polo et al., 2024; Saranathan et al., 2025; Vivek et al., 2024; Wang et al., 2025; Zhang et al., 2025).

Automatic Determination of Coreset Size Most methods in the second category assume a user-specified evaluation budget, with CER-EVAL (Wang et al., 2025) and Hsu & Shekhar (2026) as notable exceptions. The latter develop confidence sequences informed by historical model evaluations. While their coverage guarantees accommodate learned predictions, their strongest optimality result assumes oracle knowledge of question-level correctness probabilities and does not establish optimality in the practical setting. Their empirical evaluation focuses on synthetic target regimes.

Extending methods designed for user-specified coreset sizes to automatically determine the coreset size poses fundamental challenges. This task requires striving for accurate estimation across all coreset sizes, which, as we show in our experiments, is not always achievable. Furthermore, establishing a reliable stopping criterion requires confidence intervals that accurately reflect the true remaining uncertainty, a property that existing formulations often lack.

Performance Modeling and Estimation From a modeling perspective, many works that leverage reference model information rely on Item Response Theory (IRT) to infer the target model's profile (Kipnis et al., 2025; Liao et al., 2025; Polo et al., 2024). However, obtaining reliable parameter estimates with IRT requires on the order of hundreds (Jiang et al., 2026) or thousands (Kipnis et al., 2025) of reference models. Another line of work (Bowyer et al., 2026; Zhang et al., 2025) frames the problem as a regression task. PROEVAL (Huang et al., 2026) introduces a Gaussian process approach that shares similarities with our Bayesian modeling. However, we depart critically from their formulation to address automatic coreset size determination and more robustly handle the lack of information about the target model, for example, its performance on other benchmarks.

Multi-Target Setting Finally, to the best of our knowledge, the only other work that leverages performance correlations across multiple target models rather than treating them independently is Fisch et al. (2026). This approach frames model evaluation as a matrix completion problem. While the method could be run sequentially by randomly sampling the next item and model to evaluate, its confidence intervals capture only the sampling uncertainty around the mean performance, assuming the benchmark is a finite sample drawn from an infinite population of prompts. Consequently, even when all items in the finite benchmark have been evaluated, the confidence interval does not shrink

to zero. In contrast, our Bayesian credible intervals provide a direct probability statement about the true, finite-sample benchmark performance given the observed data, yielding an intuitively appealing and actionable stopping criterion.

5 EXPERIMENTS

We evaluate our approach across six benchmarks collected from three primary sources. From the Open LLM Leaderboard, we extracted scores for GPQA (Rein et al., 2024), MMLU-Pro (Wang et al., 2024; Hendrycks et al., 2021), BBH (Suzgun et al., 2023), and MuSR (Sprague et al., 2024). Beyond standard binary scores, we leveraged the leaderboard's raw log-likelihoods to obtain continuous scores via a softmax transformation. These continuous scores allow us to assess the impact of exploiting richer information and mitigate the issue of identical reference model scores. We select a representative distribution of model capabilities (see App. C.1 for details), yielding the following model counts per benchmark: GPQA: 102, MMLU-Pro: 165, BBH: 279, and MuSR: 288. From Zhang et al. (2025), we extracted binary scores for IFEval (Zhou et al., 2023) across 448 models. Finally, from HELM Lite (up to version 1.13.0), we extracted the F1 scores for the Natural QA Openbook long-answer scenario (Kwiatkowski et al., 2019) across 44 models.

5.1 SINGLE-TARGET SETTING

Baselines We compare the following estimators and sequential adaptations under the common diagnostic stopping rule. See App. C.3 for additional details.

1. **BAYESAME-RS**: A simpler version of BAYESAME that selects items at random.
2. **SEQ-APW**: An extension of the anchor points weighted method from Vivek et al. (2024), in which $\mathcal{C}$ is augmented by choosing items that maximize coverage of the space, as measured by correlations between the reference models' performances.
3. **RS-MEAN**: The simplest baseline that augments $\mathcal{C}$ by selecting items at random and estimates the full benchmark performance as the empirical average performance on $\mathcal{C}$.
4. **BAYES-AIPW**: A Bayesian ridge regression extension of AIPW (Zhang et al., 2025). Fig. 5 in App. C.4 shows that it matches or exceeds the performance of the original AIPW method.
5. **BAYES-RS-LEARN**: A Bayesian ridge regression extension of RANDOM-SAMPLING-LEARN (Zhang et al., 2025).
6. **PROEVAL**: The method in Huang et al. (2026) for the setting with no prior performance data for the target model (called New Model setting with Score Features in the original paper).
7. **IRT**: Two variants of IRT where $\mathcal{C}$ is selected randomly, depending on the score type: one for binary scores based on the 2-PL model (Polo et al., 2024) (P-IRT) and one for continuous scores based on the Beta distribution, specifically the extension of Noel & Dauvier (2007) provided in Chen et al. (2019). Item parameters are learned using the reference models' scores, and the scores $s_{\mathcal{C}}$ are used to fit the target model-specific parameters. Due to the computational cost, parameters are refitted every 50 evaluations (resulting in piecewise-constant curves in the plots).

Metrics We evaluate our framework using four metrics:

1. **RMSE LOG**: $100 \times$ Root mean squared error (RMSE) on a logarithmic scale.
2. **RMSE GAIN**: The relative improvement of a given method over the RS-MEAN baseline, calculated as: $\text{RMSE GAIN}(\text{Method}) = 20 \log_{10} (\text{RMSE}(\text{RS-MEAN})/\text{RMSE}(\text{Method}))$. A positive gain indicates a relative reduction in estimation error compared to the baseline.
3. **EVALUATION-WEIGHTED INTEGRATED RMSE**: $100 \int_0^N n' \, \text{RMSE}(n') \mathrm{d}n'$, where n' is the number of evaluated items. This metric gives greater weight to errors at larger coreset sizes.
4. **COST-ACCURACY TRADE-OFF**: The RMSE (multiplied by 100) computed at the coreset sizes dynamically determined by the ϵ_1, ϵ_2 stopping criterion, plotted against the average coreset size $\bar{n}$ across seeds. We vary $\epsilon_2 \in \{0.005, 0.0075, 0.01, 0.015, 0.02, 0.025, 0.03\}$, and fix $\epsilon_1 = 0.005$ since empirical results indicate that ϵ_2 is the primary factor governing the stopping decision.

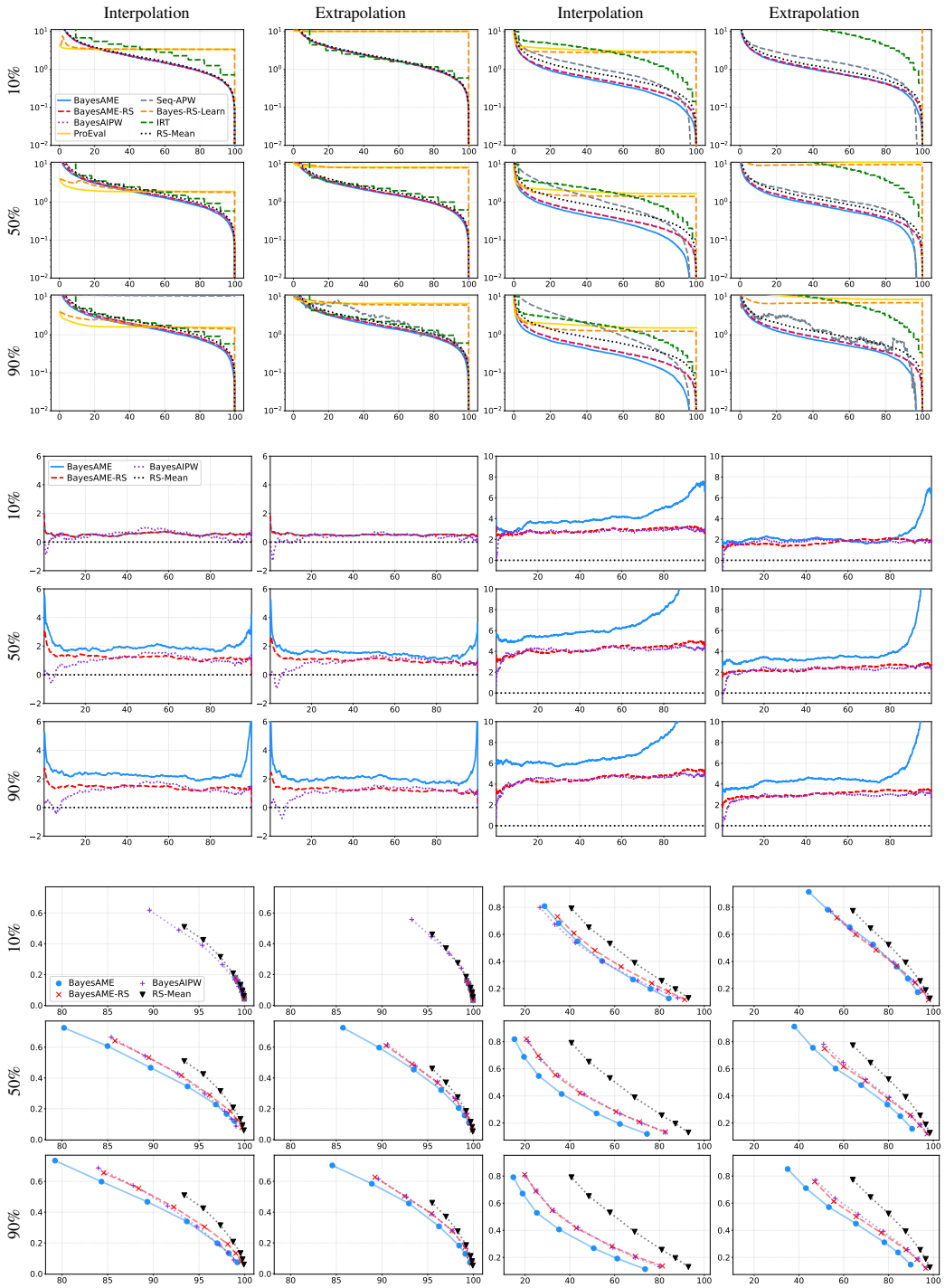

Figure 1: **Single-target Setting. GPQA** with binary scores (left two columns) and **MMLU-Pro** with continuous scores (right two columns). RMSE LOG (top three rows), RMSE GAIN (middle three rows), and COST-ACCURACY TRADE-OFF (bottom three rows) across varying proportions of reference models (10%, 50%, and 90%) for the interpolation and extrapolation regimes. The x-axis denotes the coreset size n (top three rows and middle three rows) and the average coreset size $\bar{n}$ (bottom three rows) as a percentage of the benchmark size N.

We compute the RMSE over 200 initialization seeds, accounting for both bias and variance across sampled trajectories to rigorously assess each method's estimation accuracy. The first three metrics

Table 1: **Single-target Setting.** EVALUATION-WEIGHTED INTEGRATED RMSE using 50% of the reference pool. Bold and underlined values denote the best and second-best results, respectively.

BENCHMARKS		BAYESAME	BAYESAME-RS	RS-MEAN	SEQ-APW	BAYES-AIPW	BAYES-RS-LEARN	PROEVAL	IRT
BINARY									
GPQA	Interpolation	**61.54**	66.38	76.48	866.61	67.21	99.06	96.29	89.61
	Extrapolation	**68.91**	72.65	81.73	751.08	73.41	403.29	418.11	91.40
MMLU-PRO	Interpolation	**30.18**	**30.18**	40.20	1141.06	30.68	90.59	95.56	46.38
	Extrapolation	**37.20**	**37.20**	44.89	821.40	37.69	670.20	701.84	156.33
BBH	Interpolation	**31.70**	36.35	50.42	495.01	35.54	50.63	57.82	50.95
	Extrapolation	**38.35**	42.02	52.59	448.04	41.34	448.53	600.98	158.41
MUSR	Interpolation	**47.86**	**47.86**	69.27	949.90	49.18	69.12	68.54	84.51
	Extrapolation	**54.19**	**54.19**	71.28	719.86	56.72	289.76	353.44	104.41
IFEVAL	Interpolation	**49.53**	56.14	72.45	1464.91	56.71	76.22	78.53	79.06
	Extrapolation	**58.08**	**58.08**	64.54	488.54	60.52	466.45	1017.33	212.27
CONTINUOUS									
GPQA	Interpolation	**30.58**	34.42	45.25	50.22	35.21	49.08	54.03	81.44
	Extrapolation	**40.40**	40.75	56.64	62.52	46.77	262.65	313.65	142.56
MMLU-PRO	Interpolation	**13.11**	17.10	27.80	39.99	17.19	72.66	82.62	66.44
	Extrapolation	**26.82**	31.10	40.83	47.07	31.33	482.43	570.76	311.12
BBH	Interpolation	**13.04**	17.54	35.21	49.38	18.01	26.13	40.45	61.84
	Extrapolation	**29.80**	35.57	47.10	48.50	33.64	303.09	612.49	388.23
MUSR	Interpolation	**27.86**	33.42	50.67	76.21	34.34	50.02	62.88	57.39
	Extrapolation	**36.98**	42.57	60.19	70.64	43.69	246.90	372.08	211.37
NATURAL QA OPENBOOK	Interpolation	**29.63**	**29.63**	45.91	1185.87	29.12	275.90	286.37	129.11
	Extrapolation	**26.86**	**26.86**	41.76	1044.02	27.70	226.18	262.55	127.84

evaluate performance across fixed coreset sizes $n = 1, \ldots, N$, allowing us to determine whether any method consistently outperforms the others regardless of the stopping point. In contrast, the COST-ACCURACY TRADE-OFF measures estimation error at the coreset sizes selected by the (ϵ_1, ϵ_2) stopping criterion, demonstrating which method achieves the lowest estimation error at the smallest automatically determined coreset size. We also evaluate our method using Spearman's rank correlation to assess how well the estimated performance preserves the true relative ordering of the models, see App. C.4.

Interpolation and Extrapolation Reference Setup Following Zhang et al. (2025), we evaluate interpolation and extrapolation regimes by partitioning models into reference and target sets. We additionally vary the number of reference models by sampling 10%, 50%, or 90% of the reference pool to assess its impact on performance. Detailed construction of the reference and target sets is deferred to App. C.2.

5.1.1 RESULTS

The top three rows and the middle three rows in Fig. 1 show, respectively, the RMSE LOG and RMSE GAIN for GPQA with binary scores (left two columns) and MMLU-Pro with continuous scores (right two columns), using 10%, 50%, and 90% of reference models for the interpolation and extrapolation regimes. BAYESAME achieves the lowest RMSE LOG across all coreset sizes. The gain plots compare BAYESAME, BAYESAME-RS, and BAYES-AIPW against RS-MEAN, omitting less competitive methods for clarity. Gains increase with the number of reference models, particularly for BAYESAME, suggesting that it benefits more from richer reference information. For GPQA with 10% of references, $B < N - N_0$, so BAYESAME and BAYESAME-RS coincide by construction and perform comparably to BAYES-AIPW.

The bottom three rows in Fig. 1 display the COST-ACCURACY TRADE-OFF. By providing the most accurate performance estimates for any threshold value at the lowest cost across both the interpolation and extrapolation regimes, BAYESAME establishes a strictly superior Pareto frontier.

Plots corresponding to Fig. 1 for the other benchmarks, which show similar conclusions, are provided in App. C.4. We summarize the results for 50% of reference models via the EVALUATION-WEIGHTED INTEGRATED RMSE in Table 1. The table shows that BAYESAME outperforms the other methods in nearly all cases. It also clearly quantifies the overall benefit of using richer continuous scores over binary scores. Our results demonstrate that active coreset selection becomes significantly advantageous over random coreset selection whenever reference models provide rich representations (e.g., via larger reference pools or continuous scores), and show that it is possible to outperform RS-MEAN.

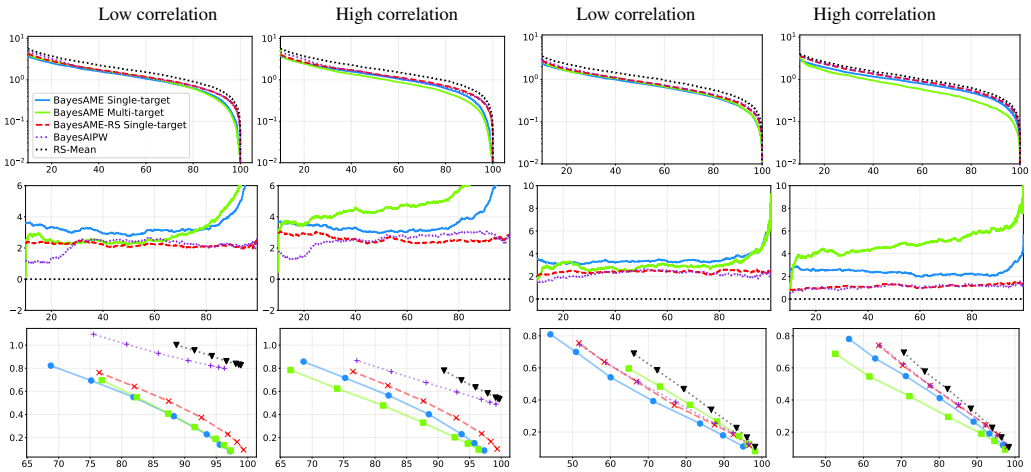

Figure 2: **Multi-target Setting. IFEval** with binary (left two columns) and **GPQA** with continuous scores (right two columns). The rows represent RMSE LOG, RMSE GAIN, and COST-ACCURACY TRADE-OFF, respectively, using 50% of reference models in the low and high correlation regimes.

5.2 MULTI-TARGET SETTING

We compare the multi-target version of BAYESAME (Section 3.2) against its single-target counterpart (Section 3.1) applied independently to each target model. To evaluate the multi-target formulation under varying degrees of correlation, we construct sets of 10 target models exhibiting either low or high correlation. Specifically, we form the low-correlation set using a random selection of 10 models, and the high-correlation set using the 10 most correlated models overall. We construct the set of reference models by sampling 50% of the reference model pool uniformly at random.

Fig. 2 shows RMSE LOG, RMSE GAIN, and COST-ACCURACY TRADE-OFF for IFEval with binary scores and GPQA with continuous scores. Additional results are given in App. C.5. The main conclusion is that jointly evaluating multiple target models provides a significant efficiency advantage when the models exhibit correlated behaviors. Specifically, in the high-correlation regime, BAYESAME MULTI-TARGET demonstrates a steeper decline in RMSE LOG and reaches a higher RMSE GAIN much earlier compared to BAYESAME SINGLE-TARGET. This is further reflected in the COST-ACCURACY TRADE-OFF plots, where BAYESAME MULTI-TARGET establishes a strictly superior Pareto frontier, achieving the same level of estimation accuracy at a notably lower coreset size across all threshold values. However, the low-correlation regime plots reveal a practical nuance of the multi-target framework. When target models do not share meaningful behavioral similarities, attempting to jointly model their latent abilities introduces a degree of negative transfer. As observed in the COST-ACCURACY TRADE-OFF plots, the multi-target approach slightly underperforms BAYESAME SINGLE-TARGET. Therefore, practitioners should apply the multi-target extension primarily when there is a reasonable prior expectation of correlated performance.

6 CONCLUSION

We introduced BAYESAME, a simple, high-performing evaluation method that automatically determines the coreset size when reliable performance estimation takes priority over efficiency, while also supporting predefined sizes. Its multi-target extension exploits performance correlations among target models to further reduce coreset sizes. Addressing recent skepticism about improvements over random selection and sample-mean estimation, our study shows that non-random selection and robust estimation can offer substantial gains when richer reference information is available. Several limitations suggest directions for future research. Our framework currently assumes scalar scores, extending it to multidimensional or unstructured evaluations is a natural next step. Incorporating model architectures, training hyperparameters, or training data distributions could also help refine the prior covariance.

AI USE STATEMENT

We acknowledge the use of Large Language Models (LLMs) for preparing this manuscript. Specifically, LLMs were used to improve grammar and clarity and generate boilerplate code snippets for our experiments and testing scripts. The authors have carefully reviewed and edited all LLM-generated outputs and take full responsibility for the final content and scientific integrity of this work.

REPRODUCIBILITY STATEMENT

Details of our experimental setup and implementation are provided in Section 5 and Appendix C. We will release the code to facilitate reproduction of our results.

REFERENCES

Mauricio A. Álvarez, Lorenzo Rosasco, and Neil D. Lawrence. Kernels for vector-valued functions: A review. *Foundations and Trends® in Machine Learning*, 4:195–266, 2012.

Andrew M. Bean, Nabeel Seedat, Shengzhuang Chen, and Jonathan Richard Schwarz. Scales++: Compute efficient evaluation subset selection with cognitive scales embeddings. *arXiv preprint arXiv:2510.26384*, 2025.

Gabrielle Berrada, Jannik Kossen, Muhammed Razzak, Freddie Bickford Smith, Yarin Gal, and Tom Rainforth. Scaling up active testing to large language models. In *Advances in Neural Information Processing Systems*, 2025.

Sam Bowyer, Acyr Locatelli, and Kris Cao. Efficient benchmarking is just feature selection and multiple regression. *arXiv preprint arXiv:2605.25773*, 2026.

Richard H. Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16:1190–1208, 1995.

Yu Chen, Telmo Silva Filho, Ricardo B. Prudencio, Tom Diethe, and Peter Flach. β^3-IRT: A new item response model and its applications. In *International Conference on Artificial Intelligence and Statistics*, pp. 1013–1021, 2019.

Adam Fisch, Daniel Deutsch, Joshua Maynez, Alekh Agarwal, Jonathan Berant, William Cohen, Amir Globerson, and Jacob Eisenstein. CollabEval: Statistically efficient collaborative model evaluation via matrix completion. *arXiv preprint arXiv:2607.05046*, 2026.

Clémentine Fourrier, Nathan Habib, Alina Lozovskaya, Konrad Szafer, and Thomas Wolf. Open LLM leaderboard v2, 2024.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021.

Chia-Yu Hsu and Shubhanshu Shekhar. Efficient sequential evaluation of large language models. *arXiv preprint arXiv:2607.17409*, 2026.

Yizheng Huang, Wenjun Zeng, Aditi Kumaresan, and Zi Wang. ProEval: Proactive failure discovery and efficient performance estimation for generative AI evaluation. In *International Conference on Machine Learning*, 2026.

Han Jiang, Sunbeom Kwon, Jinwen Luo, Ziang Xiao, and Susu Zhang. Can we trust item response theory for AI evaluation? *arXiv preprint arXiv:2607.15190*, 2026.

André G. Journel and Charles J. Huijbregts. *Mining Geostatistics*. Academic Press, 1978.

Alex Kipnis, Konstantinos Voudouris, Luca M. Schulze Buschoff, and Eric Schulz. metabench - A sparse benchmark of reasoning and knowledge in large language models. In *International Conference on Learning Representations*, 2025.

Jannik Kossen, Sebastian Farquhar, Yarin Gal, and Tom Rainforth. Active testing: Sample-efficient model evaluation. In *International Conference on Machine Learning*, 2021.

Alex Kulesza and Ben Taskar. k-DPPs: Fixed-size determinantal point processes. In *International Conference on Machine Learning*, pp. 1193–1200, 2011.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466, 2019.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Cosgrove, Christopher D Manning, Christopher Re, Diana Acosta-Navas, Drew A. Hudson, Eric Zelikman, Esin Durmus, Faisal Ladhak, Frieda Rong, Hongyu Ren, Huaxiu Yao, Jue WANG, Keshav Santhanam, Laurel Orr, Lucia Zheng, Mert Yuksekgonul, Mirac Suzgun, Nathan Kim, Neel Guha, Niladri S. Chatterji, Omar Khattab, Peter Henderson, Qian Huang, Ryan Andrew Chi, Sang Michael Xie, Shibani Santurkar, Surya Ganguli, Tatsunori Hashimoto, Thomas Icard, Tianyi Zhang, Vishrav Chaudhary, William Wang, Xuechen Li, Yifan Mai, Yuhui Zhang, and Yuta Koreeda. Holistic evaluation of language models. *Transactions on Machine Learning Research*, 2023.

Lele Liao, Qile Zhang, Ruofan Wu, and Guanhua Fang. Toward a unified framework for data-efficient evaluation of large language models. *arXiv preprint arXiv:2510.04051*, 2025.

Zeli Liu, Jiancheng Zhang, Cong Liu, and Yinglun Zhu. Active testing of large language models via approximate Neyman allocation. *arXiv preprint arXiv:2605.10075*, 2026.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

David J. C. MacKay. The evidence framework applied to classification networks. *Neural Computation*, 4(5):720–736, 1992.

Yvonnick Noel and Bruno Dauvier. A Beta item response model for continuous bounded responses. *Applied Psychological Measurement*, 31(1):47–73, 2007.

Michael Osborne, Roman Garnett, Stephen Roberts, Christopher Hart, Suzanne Aigrain, and Neale Gibson. Bayesian quadrature for ratios. In *International Conference on Artificial Intelligence and Statistics*, pp. 832–840, 2012.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12: 2825–2830, 2011.

Yotam Perlitz, Elron Bandel, Ariel Gera, Ofir Arviv, Liat Ein-Dor, Eyal Shnarch, Noam Slonim, Michal Shmueli-Scheuer, and Leshem Choshen. Efficient benchmarking (of language models). In *Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 2519–2536, 2024.

Felipe Maia Polo, Lucas Weber, Leshem Choshen, Yuekai Sun, Gongjun Xu, and Mikhail Yurochkin. tinyBenchmarks: evaluating LLMs with fewer examples. In *International Conference on Machine Learning*, 2024.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. GPQA: A graduate-level google-proof Q&A benchmark. In *First Conference on Language Modeling*, 2024.

Gayathri Saranathan, Cong Xu, Mahammad Parwez Alam, Tarun Kumar, Martin Foltin, Soon Yee Wong, and Suparna Bhattacharya. SubLIME: Subset selection via rank correlation prediction for data-efficient LLM evaluation. In *Annual Meeting of the Association for Computational Linguistics*, pp. 30572–30593, 2025.

Zayne Sprague, Xi Ye, Kaj Bostrom, Swarat Chaudhuri, and Greg Durrett. MuSR: Testing the limits of chain-of-thought with multistep soft reasoning. In *International Conference on Learning Representations*, 2024.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and Jason Wei. Challenging BIG-Bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics*, pp. 13003–13051, 2023.

Rajan Vivek, Kawin Ethayarajh, Diyi Yang, and Douwe Kiela. Anchor points: Benchmarking models with much fewer examples. In *Conference of the European Chapter of the Association for Computational Linguistics*, pp. 1576–1601, 2024.

Ganghua Wang, Zhaorun Chen, Bo Li, and Haifeng Xu. Cer-Eval: Certifiable and cost-efficient evaluation framework for LLMs. *arXiv preprint arXiv:2505.03814*, 2025.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. MMLU-Pro: A more robust and challenging multi-task language understanding benchmark. In *Advances in Neural Information Processing Systems (Track on Datasets and Benchmarks)*, 2024.

Guanhua Zhang, Florian E. Dorner, and Moritz Hardt. How benchmark prediction from fewer data misses the mark. In *Advances in Neural Information Processing Systems*, 2025.

Hongyu Zhao, Ming Li, Lichao Sun, and Tianyi Zhou. BenTo: Benchmark reduction with in-context transferability. In *International Conference on Learning Representations*, 2025.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

APPENDIX

A BAYESAME DETAILS

A.1 POSTERIOR DISTRIBUTION COMPUTATION

In this section, we show that conditioning on the coreset scores $s_{\mathcal{C}}$ is equivalent to conditioning on the bucket-wise averages $\bar{s}_{\mathcal{C}_{1:B}}$. In particular, we have that $p(\theta \mid S_{\mathcal{C}} = s_{\mathcal{C}}) = p(\theta \mid \bar{S}_{\mathcal{C}_{1:B}} = \bar{s}_{\mathcal{C}_{1:B}}) = \mathcal{N}(\mu^{\theta}_{|\mathcal{C}}, \Sigma^{\theta}_{|\mathcal{C}})$. We derive below two equivalent expressions for $\mu^{\theta}_{|\mathcal{C}}$ and $\Sigma^{\theta}_{|\mathcal{C}}$, which we leverage to optimize the cost of computing the posterior.

Let $\mathcal{J} = \{b : n_b > 0\}$, $m = |\mathcal{J}|$ and $\mathcal{C}_b$ denote the subset of n_b items in $\mathcal{C}$ that are in bucket b. For $b \in \mathcal{J}$, define

$$\bar{S}_{\mathcal{C}_b} = \tfrac{1}{n_b} \textstyle\sum_{i \in \mathcal{C}_b} S_i, \quad \bar{s}_{\mathcal{C}_b} = \tfrac{1}{n_b} \textstyle\sum_{i \in \mathcal{C}_b} s_i.$$

Write $\bar{S}_{\mathcal{C}_{1:B}} = (\bar{S}_{\mathcal{C}_b})_{b \in \mathcal{J}}$ and $\bar{s}_{\mathcal{C}_{1:B}} = (\bar{s}_{\mathcal{C}_b})_{b \in \mathcal{J}}$. Recall that $\bar{S}_{\mathcal{C}_b} \mid \theta_b \sim \mathcal{N}\big(\theta_b, \frac{\sigma^2 N_b}{n_b}\big)$. For the observed scores, we have:

$$\begin{aligned}
p(s_{\mathcal{C}_b} \mid \theta_b) &\propto \exp\Big(-\tfrac{1}{2\sigma^2 N_b} \textstyle\sum_{i \in \mathcal{C}_b} (s_i - \theta_b)^2\Big) \\
&= \exp\Big(-\tfrac{1}{2\sigma^2 N_b} \textstyle\sum_{i \in \mathcal{C}_b} ((s_i - \bar{s}_{\mathcal{C}_b}) + (\bar{s}_{\mathcal{C}_b} - \theta_b))^2\Big) \\
&= \exp\Big(-\tfrac{1}{2\sigma^2 N_b} \big(\textstyle\sum_{i \in \mathcal{C}_b} (s_i - \bar{s}_{\mathcal{C}_b})^2 + n_b (\bar{s}_{\mathcal{C}_b} - \theta_b)^2\big)\Big) \\
&= \exp\Big(-\tfrac{1}{2\sigma^2 N_b} \textstyle\sum_{i \in \mathcal{C}_b} (s_i - \bar{s}_{\mathcal{C}_b})^2\Big) \exp\Big(-\tfrac{n_b}{2\sigma^2 N_b} (\bar{s}_{\mathcal{C}_b} - \theta_b)^2\Big) \\
&\propto \exp\Big(-\tfrac{n_b}{2\sigma^2 N_b} (\bar{s}_{\mathcal{C}_b} - \theta_b)^2\Big) \\
&\propto p(\bar{s}_{\mathcal{C}_b} \mid \theta_b),
\end{aligned}$$

where the proportionality is with respect to θ_b. This results into:

$$\begin{aligned}
p(\theta \mid s_{\mathcal{C}}) &\propto p(\theta) p(s_{\mathcal{C}} \mid \theta) = p(\theta) \textstyle\prod_{b=1;\, n_b > 0}^{B} p(s_{\mathcal{C}_b} \mid \theta_b) \propto p(\theta) \prod_{b=1;\, n_b > 0}^{B} p(\bar{s}_{\mathcal{C}_b} \mid \theta_b) \\
&= p(\theta) p(\bar{s}_{\mathcal{C}_{1:B}} \mid \theta) \propto p(\theta \mid \bar{s}_{\mathcal{C}_{1:B}}),
\end{aligned}$$

which establishes that $p(\theta \mid S_{\mathcal{C}} = s_{\mathcal{C}}) = p(\theta \mid \bar{S}_{\mathcal{C}_{1:B}} = \bar{s}_{\mathcal{C}_{1:B}})$.

Having shown this equivalence, we now derive the explicit form for $p(\theta \mid S_{\mathcal{C}} = s_{\mathcal{C}}) = p(\theta \mid \bar{S}_{\mathcal{C}_{1:B}} = \bar{s}_{\mathcal{C}_{1:B}})$ through the Gaussian conditioning rule, which states that, for a joint distribution $p(x_1, x_2)$, the conditional distribution $p(x_1 \mid x_2)$ has mean $\mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(x_2 - \mu_2)$ and covariance $\Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$.

Let $H_{\mathcal{C}}$ and $D_{\mathcal{C}}$ denote the submatrices of H and D formed by the rows corresponding to $\mathcal{C}$ for H and both the rows and columns corresponding to $\mathcal{C}$ for D. The joint distribution of $(\theta, S_{\mathcal{C}})$ can be written as:

$$\begin{bmatrix} \theta \\ S_{\mathcal{C}} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu^{\theta} \\ H_{\mathcal{C}} \mu^{\theta} \end{bmatrix}, \begin{bmatrix} \Sigma^{\theta} & \Sigma^{\theta} H_{\mathcal{C}}^{\top} \\ H_{\mathcal{C}} \Sigma^{\theta} & H_{\mathcal{C}} \Sigma^{\theta} H_{\mathcal{C}}^{\top} + D_{\mathcal{C}} \end{bmatrix} \right).$$

Applying the Gaussian conditioning rule, we obtain:

$$\begin{aligned}
\mu^{\theta}_{|\mathcal{C}} &= \mu^{\theta} + \Sigma^{\theta} H_{\mathcal{C}}^{\top} (H_{\mathcal{C}} \Sigma^{\theta} H_{\mathcal{C}}^{\top} + D_{\mathcal{C}})^{-1} (s_{\mathcal{C}} - H_{\mathcal{C}} \mu^{\theta}), \\
\Sigma^{\theta}_{|\mathcal{C}} &= \Sigma^{\theta} - \Sigma^{\theta} H_{\mathcal{C}}^{\top} (H_{\mathcal{C}} \Sigma^{\theta} H_{\mathcal{C}}^{\top} + D_{\mathcal{C}})^{-1} H_{\mathcal{C}} \Sigma^{\theta}.
\end{aligned} \tag{7}$$

Let $P \in \mathbb{R}^{m \times B}$ select the coordinates in $\mathcal{J}$, in the same order as the bucket-average vector. Let Δ be a $m \times m$ diagonal noise matrix with entries $\Delta_{b,b} = \sigma^2 N_b / n_b$. The joint distribution of $(\theta, \bar{S}_{\mathcal{C}_{1:B}})$ can be written as:

$$\begin{bmatrix} \theta \\ \bar{S}_{\mathcal{C}_{1:B}} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu^{\theta} \\ P \mu^{\theta} \end{bmatrix}, \begin{bmatrix} \Sigma^{\theta} & \Sigma^{\theta} P^{\top} \\ P \Sigma^{\theta} & P \Sigma^{\theta} P^{\top} + \Delta \end{bmatrix} \right).$$

Applying the Gaussian conditioning rule, we obtain:

$$\mu^{\theta}_{|\mathcal{C}} = \mu^{\theta} + \Sigma^{\theta} P^{\top} (P\Sigma^{\theta} P^{\top} + \Delta)^{-1} (\bar{s}_{\mathcal{C}_{1:B}} - P\mu^{\theta}),$$
$$\Sigma^{\theta}_{|\mathcal{C}} = \Sigma^{\theta} - \Sigma^{\theta} P^{\top} (P\Sigma^{\theta} P^{\top} + \Delta)^{-1} P\Sigma^{\theta}. \tag{8}$$

To reduce the computational cost, we use Eq. (7) when $n < B$ and Eq. (8) when $n \geq B$. We employ Cholesky updates across both regimes, leveraging the fact that the prior covariance Σ^{θ} remains fixed between hyperparameter updates, which occur every F iterations. This yields a computational complexity of $\mathcal{O}(\min(|\mathcal{C}|, B)^2)$ per iteration between hyperparameter refits.

A.2 Performance Estimator

In this section, we discuss the estimator $\hat{R}_{CV}$ (Eq. (4)) that we use in the non-unique reference regime $B < N - N_0$. To simplify the exposition, we adopt an item-level, rather than bucket-level, description. Specifically, for an item (x_i, y_i) belonging to bucket b, we use $\mu_{i|\mathcal{C}}$ to denote $\mathbb{E}[\theta_b \,|\, S_{\mathcal{C}} = s_{\mathcal{C}}]$. We can rewrite Eq. (4) at the item-level as:

$$\hat{R}_{\text{CV}} = \tfrac{1}{n} \textstyle\sum_{i \in \mathcal{C}} \left(s_i - \mu_{i|\mathcal{C}}\right) + \tfrac{1}{N} \sum_{j=1}^{N} \mu_{j|\mathcal{C}}.$$

This estimator can be seen as an in-sample proxy of the following control variate estimator that uses out-of-sample posteriors:

$$\hat{R}_{\text{CVO}} = \tfrac{1}{n} \textstyle\sum_{i \in \mathcal{C}} \left(s_i - \mu_{i|\mathcal{C}\setminus\{i\}} + \tfrac{1}{N} \sum_{j=1}^{N} \mu_{j|\mathcal{C}\setminus\{i\}}\right).$$

Note that the in-sample proxy estimator avoids the computational bottleneck of recomputing the posterior for every held-out item.

For a fixed sample size $n \geq 1$ and conditional on the fixed benchmark scores $s_1, \ldots, s_N$, $\hat{R}_{\text{CVO}}$ is unbiased with respect to independent uniform sampling with replacement as shown below. Let $\mathcal{C} = (i_1, \ldots, i_n)$ be constructed by sampling items uniformly at random with replacement, such that $i_k \sim U(1, \ldots, N)$ independently. Here $\mathcal{C}$ is an indexed sequence: sums count multiplicities and $\mathcal{C} \setminus \{i_k\}$ means deletion of the k-th sampled occurrence only, retaining any other occurrences of the same item. Each held-out predictor must depend only on the remaining observations and information fixed independently of the sample. $\hat{R}_{\text{CVO}}$ can be expressed as:

$$\hat{R}_{\text{CVO}} = \tfrac{1}{n} \textstyle\sum_{k=1}^{n} \left(s_{i_k} - \mu_{i_k|\mathcal{C}\setminus\{i_k\}} + \tfrac{1}{N} \sum_{j=1}^{N} \mu_{j|\mathcal{C}\setminus\{i_k\}}\right).$$

Taking the unconditional expectation with respect to the random selection of the coreset $\mathcal{C}$ yields:

$$\mathbb{E}[\hat{R}_{\text{CVO}}] = \tfrac{1}{n} \textstyle\sum_{k=1}^{n} \mathbb{E}[s_{i_k}] - \tfrac{1}{n} \sum_{k=1}^{n} \left(\mathbb{E}\left[\mu_{i_k|\mathcal{C}\setminus\{i_k\}}\right] - \tfrac{1}{N} \sum_{j=1}^{N} \mathbb{E}\left[\mu_{j|\mathcal{C}\setminus\{i_k\}}\right]\right). \tag{9}$$

We first focus on the term $\mathbb{E}\left[\mu_{i_k|\mathcal{C}\setminus\{i_k\}}\right]$. Using the Law of Total Expectation, we have:

$$\mathbb{E}\left[\mu_{i_k|\mathcal{C}\setminus\{i_k\}}\right] = \mathbb{E}_{\mathcal{C}\setminus\{i_k\}}\left[\mathbb{E}\left[\mu_{i_k|\mathcal{C}\setminus\{i_k\}} | \mathcal{C} \setminus \{i_k\}\right]\right].$$

Because i_k is drawn independently from the remaining coreset items $\mathcal{C} \setminus \{i_k\}$, the inner expectation over the uniform draw of i_k is exactly the population mean:

$$\mathbb{E}\left[\mu_{i_k|\mathcal{C}\setminus\{i_k\}}\right] = \mathbb{E}_{\mathcal{C}\setminus\{i_k\}}\left[\tfrac{1}{N} \textstyle\sum_{j=1}^{N} \mu_{j|\mathcal{C}\setminus\{i_k\}}\right] = \tfrac{1}{N} \sum_{j=1}^{N} \mathbb{E}\left[\mu_{j|\mathcal{C}\setminus\{i_k\}}\right].$$

Plugging this into Eq. (9), the second term cancels out, resulting in:

$$\begin{aligned}\mathbb{E}[\hat{R}_{\text{CVO}}] &= \tfrac{1}{n} \textstyle\sum_{k=1}^{n} \mathbb{E}[s_{i_k}] = \tfrac{1}{n} \sum_{k=1}^{n} \mathbb{E}_{\mathcal{C}\setminus\{i_k\}}[\mathbb{E}[s_{i_k} \,|\, \mathcal{C} \setminus \{i_k\}]] \\ &= \tfrac{1}{n} \textstyle\sum_{k=1}^{n} \mathbb{E}_{\mathcal{C}\setminus\{i_k\}}\left[\tfrac{1}{N} \sum_{j=1}^{N} s_j\right] = \tfrac{1}{N} \sum_{j=1}^{N} s_j = R^{\star}.\end{aligned}$$

Remark: In practice, the coreset is sampled without replacement. This introduces dependence between i_k and $\mathcal{C} \setminus \{i_k\}$, so the preceding argument no longer guarantees unbiasedness.

A.3 Stopping Criterion

Recall that, to monitor posterior uncertainty about the random variable R, we consider the width of the 95% credible interval for R, which is governed by $\mathrm{Var}(R \mid s_{\mathcal{C}})$. Crucially, this quantity is not tied to a particular choice of point estimator. This distinction is important in the non-unique reference regime, $B < N - N_0$, where we use the control-variate estimator $\hat{R}_{\mathrm{CV}}$ in Eq. (4). Our stopping criterion deliberately separates these two roles: the stability condition (i) is applied to the reported estimator $\hat{R}_{\mathrm{CV}}$, whereas the credible-interval condition (ii) is applied to the target quantity R.

A natural alternative would be to define a random variable $R_{\mathrm{CV}} = \frac{1}{n}\sum_{i\in\mathcal{C}} s_i + \sum_{b=1}^{B}\left(\frac{N_b}{N} - \frac{n_b}{n}\right)\theta_b$ and use $\mathrm{Var}(R_{\mathrm{CV}} \mid s_{\mathcal{C}})$ in the stopping criterion. However, this variance is not reliable as a measure of uncertainty about R since it vanishes whenever $N_b/N = n_b/n$, which occurs, e.g., when there is only one bucket. Another possible estimator-specific diagnostic is the posterior MSE, $\mathbb{E}[(R - \hat{R}_{\mathrm{CV}})^2 \mid s_C] = \mathrm{Var}(R \mid s_{\mathcal{C}}) + (\hat{R}_{\mathrm{Bayes}} - \hat{R}_{\mathrm{CV}})^2$. This is a valid posterior risk criterion under squared-error loss: the first term captures residual uncertainty about R, while the second captures the squared difference between $\hat{R}_{\mathrm{CV}}$ and the posterior mean. We use $\mathrm{Var}(R \mid s_{\mathcal{C}})$ as the default uncertainty measure because it preserves a direct credible-interval interpretation for the benchmark performance and avoids conflating posterior uncertainty with the deliberate correction introduced by the control-variate estimator.

A.4 Hyperparameter Optimization

Single-target Setting We tune the hyperparameters α and β, which govern the prior covariance Σ^{θ}, by minimizing the negative log-marginal likelihood $\mathcal{L}(\alpha, \beta)$ of the coreset scores. As established in Appendix A.1, the conditional likelihoods $p(s_{\mathcal{C}} \mid \theta)$ and $p(\bar{s}_{\mathcal{C}_{1:B}} \mid \theta)$ are proportional up to a constant independent of θ. As a result, their negative log-marginal likelihoods differ only by a constant independent of α and β. Therefore, to evaluate the objective efficiently, we define $\mathcal{L}(\alpha, \beta)$ as:

$$\mathcal{L}(\alpha,\beta) = \begin{cases} \frac{1}{2}(s_{\mathcal{C}} - H_{\mathcal{C}}\mu^{\theta})^{\top}(H_{\mathcal{C}}\Sigma^{\theta}H_{\mathcal{C}}^{\top} + D_{\mathcal{C}})^{-1}(s_{\mathcal{C}} - H_{\mathcal{C}}\mu^{\theta}) + \frac{1}{2}\log|H_{\mathcal{C}}\Sigma^{\theta}H_{\mathcal{C}}^{\top} + D_{\mathcal{C}}| & \text{if } n < B \\ \frac{1}{2}(\bar{s}_{\mathcal{C}_{1:B}} - P\mu^{\theta})^{\top}(P\Sigma^{\theta}P^{\intercal} + \Delta)^{-1}(\bar{s}_{\mathcal{C}_{1:B}} - P\mu^{\theta}) + \frac{1}{2}\log|P\Sigma^{\theta}P^{\intercal} + \Delta| & \text{if } n \geq B \end{cases} \tag{10}$$

For additional efficiency and to maintain numerical stability, we employ a Cholesky decomposition of the respective covariance matrix. We perform this minimization via an L-BFGS-B optimizer (Byrd et al., 1995), constraining $\alpha, \beta \in [10^{-3}, 10]$ to ensure the covariance matrix remains positive definite and to prevent degenerate length-scales.

Multi-target Setting We tune the hyperparameters α_l and β_l which govern the prior covariance $\Sigma^{\theta,l}$, as well as the model-specific weights $\{w_l^t\}_{t=1}^{K_t}$ for each $l = 1, \ldots, L$. Additionally, we learn the noise variance σ^2 instead of keeping it fixed as for the single-target setting. This turns minimizing the negative log-marginal likelihood into a highly non-convex, joint optimization problem. Consequently, instead of L-BFGS-B, we employ AdamW (Loshchilov & Hutter, 2019). AdamW leverages momentum to successfully navigate complex loss surfaces and provides adaptive, per-parameter learning rates, which are essential for handling parameters that operate on fundamentally different scales. Because the hyperparameters naturally stabilize as the coreset grows, we deploy an adaptive computational schedule that executes 500 gradient steps during the first 100 active learning iterations and 100 thereafter to significantly reduce computational overhead. To ensure numerical stability throughout the training process, we clip the values of α_l, β_l, and σ^2 after each update, constraining $\alpha_l, \beta_l \in [10^{-3}, 10]$ and $\sigma^2 \in [10^{-4}, 1]$. The weights are left unconstrained.

B BayesAME Single-target Variants

B.1 Batch Selection Strategies

We extend the approach from the main text by introducing selection strategies that seek an optimal batch of items A, rather than a single item. We focus on the nearly unique reference regime $B \geq N - N_0$. To simplify the presentation, we denote $\Sigma_{|\mathcal{C}} = H\Sigma^{\theta}_{|\mathcal{C}}H^{\top} + D$, which in the case $B = N$ simplifies to $\Sigma_{|\mathcal{C}} = \Sigma^{\theta}_{|\mathcal{C}} + \sigma^2 I_N$.

The extension of the expected information gain Eq. (6) from one item to a batch of items A is given by:

$$\begin{aligned}\alpha_{\mathrm{IG}}(A) &= \mathcal{H}(R \mid s_{\mathcal{C}}) - \mathbb{E}_{S_A}[\mathcal{H}(R \mid s_{\mathcal{C}}, S_A)] \\ &= -\tfrac{1}{2}\log\left(1 - \tfrac{1}{N^2 \operatorname{Var}(R \mid s_{\mathcal{C}})}(\Sigma_{A,U|\mathcal{C}}\mathbf{1}_{|U|})^\top \Sigma_{A,A|\mathcal{C}}^{-1}(\Sigma_{A,U|\mathcal{C}}\mathbf{1}_{|U|})\right),\end{aligned}$$

where $\Sigma_{A,U|\mathcal{C}}$ denotes the submatrix of $\Sigma_{|\mathcal{C}}$ of rows and columns corresponding to A and the unevaluated set $U = E \setminus \mathcal{C}$, respectively, and $\mathbf{1}_{|U|}$ is a $|U|$-dimensional vector of ones. A fundamental property of Gaussian conditioning where the noise is not a function of the score value implies that $\Sigma_{|\mathcal{C}}$ depends only on the score indexes rather than their actual values. This means that batch selection strategies simplify to a purely combinatorial problem of identifying an optimal batch of indices. Once this optimal batch is identified, the specific sequence in which these items are evaluated is arbitrary.

Because the expected information gain is strictly monotonically increasing with respect to the variance reduction, maximizing $\alpha_{\mathrm{IG}}(A)$ is equivalent to maximizing the variance reduction term:

$$\beta(A) = \tfrac{1}{N^2}(\Sigma_{A,U|\mathcal{C}}\mathbf{1}_{|U|})^\top \Sigma_{A,A|\mathcal{C}}^{-1}(\Sigma_{A,U|\mathcal{C}}\mathbf{1}_{|U|}). \tag{11}$$

Therefore, for a batch A of specified size $|A| = k$, where $1 \le k \le |U|$, we obtain the following optimization problem:

$$A^\star = \operatorname{argmax}_{A \subseteq U, |A|=k} \beta(A). \tag{12}$$

Batch Size $|A| = 2$ When $|A| = 2$, we can derive a closed-form expression of $\beta(A = \{i, j\})$ that can be efficiently vectorized:

$$\beta(\{i,j\}) = \tfrac{1}{N^2}\begin{pmatrix} T_i & T_j \end{pmatrix}\begin{pmatrix} \Sigma_{i,i|\mathcal{C}} & \Sigma_{i,j|\mathcal{C}} \\ \Sigma_{i,j|\mathcal{C}} & \Sigma_{j,j|\mathcal{C}} \end{pmatrix}^{-1}\begin{pmatrix} T_i \\ T_j \end{pmatrix} = \tfrac{1}{N^2}\tfrac{T_i^2\Sigma_{j,j|\mathcal{C}} + T_j^2\Sigma_{i,i|\mathcal{C}} - 2T_iT_j\Sigma_{i,j|\mathcal{C}}}{\Sigma_{i,i|\mathcal{C}}\Sigma_{j,j|\mathcal{C}} - \Sigma_{i,j|\mathcal{C}}^2},$$

where $T_i = \sum_{k \in U}\Sigma_{i,k|\mathcal{C}}$ and similarly for T_j.

Batch Size $|A| > 2$ As the batch size $|A|$ grows, an exhaustive search over all $\binom{|U|}{|A|}$ subsets requires $\mathcal{O}(|U|^{|A|}|A|^3)$ operations, which quickly becomes computationally infeasible even for moderate $|U|$. We address this by replacing the exhaustive search with a heuristic search that scales linearly with $|U|$. Specifically, we use heuristic warm-start strategies to identify initial candidate batches, followed by a combination of local search and Sequential Monte Carlo (SMC). Coupling local search with SMC can reduce the risk of getting stuck in local optima: we maintain a population of candidate batches, apply local search to each independently, and resample them according to their fitness $\beta(A)$. We use the following warm-start strategies:

- **LARS**: Solve an ℓ_1-penalized continuous relaxation of Eq. (12). In particular, for $w \in \mathbb{R}^N$ with $w_{\mathcal{C}} = \mathbf{0}$ define the following function:

  $$g(w) = \tfrac{1}{2}\, w^\top \Sigma_{|\mathcal{C}}\, w - w^\top v,$$

  where $v = \Sigma_{E,U|\mathcal{C}}\mathbf{1}_{|U|}$. Fix a subset A of U and let $w_{A^c} = \mathbf{0}$. The gradient of g with respect to w_A is

  $$\nabla_{w_A} g = \Sigma_{A,A|\mathcal{C}}\, w_A - v_A,$$

  where $v_A = \Sigma_{A,U|\mathcal{C}}\mathbf{1}_{|U|}$. Setting the gradient to zero yields the optimizer $w_A^\star = \Sigma_{A,A|\mathcal{C}}^{-1} v_A$. Substituting $w_A^\star$ into g gives

  $$g(w_A^\star) = -\tfrac{1}{2} v_A^\top \Sigma_{A,A|\mathcal{C}}^{-1} v_A = -\tfrac{N^2}{2}\beta(A).$$

  Therefore, maximizing $\beta(A)$ over all subsets A of size k is equivalent to solving the ℓ_0-constrained quadratic program

  $$\min_{w \in \mathbb{R}^N}\ \tfrac{1}{2}\, w^\top \Sigma_{|\mathcal{C}} w - w^\top v \quad \text{subject to} \quad \|w\|_0 \le k,\ w_{\mathcal{C}} = \mathbf{0}.$$

  Since this problem is NP-hard in general, we instead consider the standard ℓ_1 relaxation

  $$\min_{w \in \mathbb{R}^N}\ \tfrac{1}{2}\, w^\top \Sigma_{|\mathcal{C}} w - w^\top v + \lambda\|w\|_1, \quad \text{subject to} \quad w_{\mathcal{C}} = \mathbf{0}, \quad \lambda > 0.$$

  To solve this, we apply the Lasso modification of the least angle regression (LARS) algorithm to efficiently extract an initial active set of $|A|$ items.

- **DPP**: Sample multiple diverse initial batches using a k-determinantal point process (k-DPP) (Kulesza & Taskar, 2011) restricted to size $|A|$, using the posterior covariance matrix of the candidates as the likelihood kernel.
- **CLUST**: Perform K-means clustering on the reference model score vectors ϕ_i to partition the candidate pool into $|A|$ distinct clusters. Generate multiple initial sets by uniformly sampling one item from each cluster.

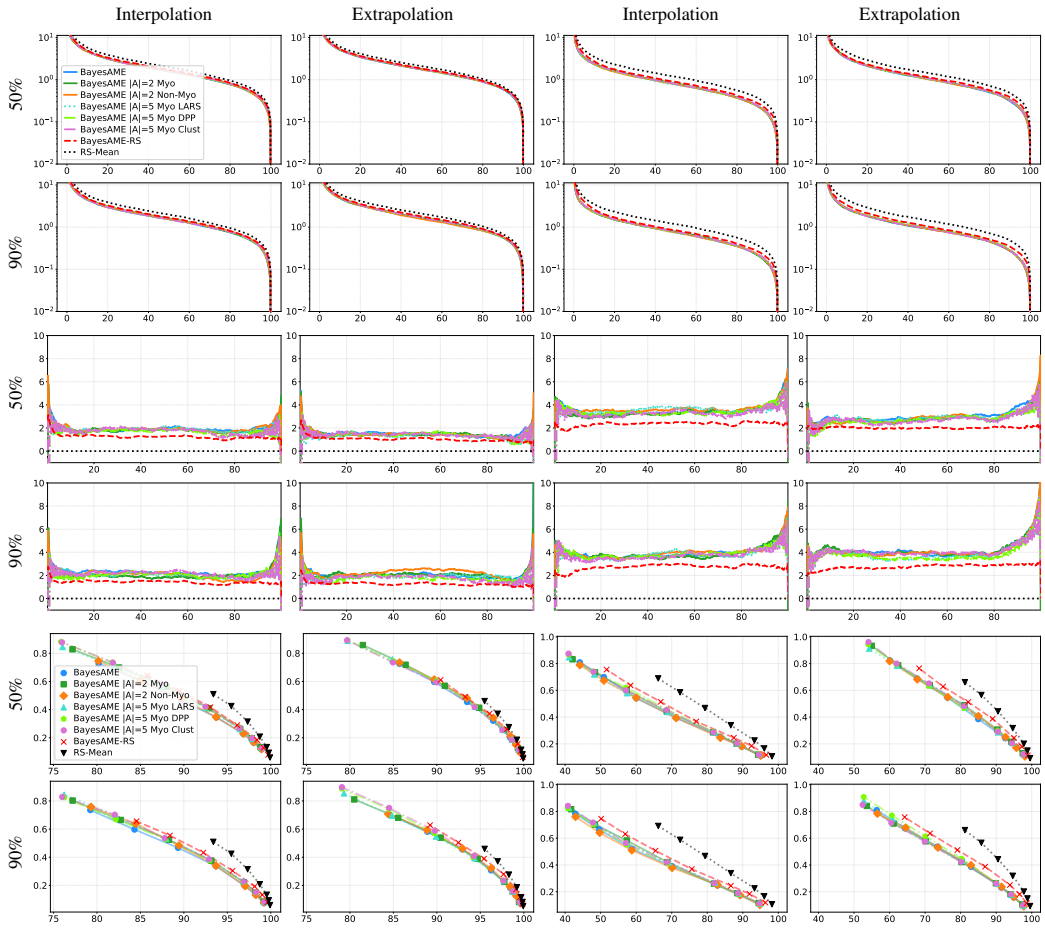

Figure 3: **Single-target Setting. GPQA** with binary scores (left two columns) and continuous scores (right two columns). RMSE LOG (top two rows) and RMSE GAIN (central two rows), and COST-ACCURACY TRADE-OFF (bottom two rows) across 50%, and 90% of reference models (percentages with unique reference scores $B = N$).

Crucially, during the local search phase, evaluating a neighbor where only a single item in the candidate batch is swapped can be computed highly efficiently using rank-2 updates. Suppose we wish to recompute the inverse in Eq. (11) after a swap. Let $M_A = \Sigma_{A,A|\mathcal{C}}$ be the original $|A| \times |A|$ matrix and $M'_A = \Sigma_{\tilde{A},\tilde{A}|\mathcal{C}}$ be the updated matrix. If the m-th item is swapped from A to $\tilde{A}$, then M_A and M'_A differ only by their m-th row and column. Let $e_m \in \mathbb{R}^{|A|}$ denote the m-th standard basis vector (a vector of zeros with a 1 at the m-th coordinate). Let c_m be the m-th column of M_A and $\tilde{c}_m$ be the corresponding column in M'_A. Defining the difference vector $d = \tilde{c}_m - c_m$, we can express the update as:

$$M'_A = M_A + de_m^\top + e_m d^\top - d_m e_m e_m^\top,$$

where d_m denotes the m-th coordinate of d. This can be factorized into $M'_A = M_A + UV^\top$, where

$$U = \begin{bmatrix} e_m & d - \frac{1}{2} d_m e_m \end{bmatrix}, \quad V = \begin{bmatrix} d - \frac{1}{2} d_m e_m & e_m \end{bmatrix}.$$

Table 2: EVALUATION-WEIGHTED INTEGRATED RMSE using 50% of the reference pool. A dash denotes that we are in a non-unique reference regime. Therefore, selection defaults to random and the entry is omitted. Bold and underlined values denote the best and second-best results, respectively.

BENCHMARKS		BAYESAME	BAYESAME-RS	BAYESAME $\lvert A\rvert = 2$ MYO	BAYESAME $\lvert A\rvert = 2$ NON-MYO	BAYES-AIPW
BINARY						
GPQA	Interpolation	<u>61.54</u>	66.38	62.08	**61.51**	67.21
	Extrapolation	**68.91**	72.65	<u>69.43</u>	69.50	73.41
MMLU-PRO	Interpolation	**30.18**	**30.18**	-	-	<u>30.68</u>
	Extrapolation	**37.20**	**37.20**	-	-	<u>37.69</u>
BBH	Interpolation	31.70	36.35	**30.51**	<u>30.98</u>	35.54
	Extrapolation	**38.35**	42.02	<u>38.47</u>	38.66	41.34
MUSR	Interpolation	**47.86**	**47.86**	-	-	<u>49.18</u>
	Extrapolation	**54.19**	**54.19**	-	-	<u>56.72</u>
IFEVAL	Interpolation	**49.53**	56.14	<u>49.99</u>	50.00	56.71
	Extrapolation	**58.08**	**58.08**	-	-	<u>60.52</u>
CONTINUOUS						
GPQA	Interpolation	<u>30.58</u>	34.42	30.84	**29.88**	35.21
	Extrapolation	**40.40**	<u>40.75</u>	41.17	40.76	46.77
MMLU-PRO	Interpolation	**13.11**	17.10	<u>13.14</u>	13.32	17.19
	Extrapolation	26.82	31.10	**26.24**	<u>26.60</u>	31.33
BBH	Interpolation	<u>13.04</u>	17.54	**12.81**	13.13	18.01
	Extrapolation	**29.80**	35.57	30.62	<u>30.49</u>	33.64
MUSR	Interpolation	<u>27.86</u>	33.42	27.94	**27.43**	34.34
	Extrapolation	36.98	42.57	**35.70**	<u>36.57</u>	43.69
NATURAL QA OPENBOOK	Interpolation	<u>29.63</u>	<u>29.63</u>	-	-	**29.12**
	Extrapolation	**26.86**	**26.86**	-	-	<u>27.70</u>

This rank-2 factorization allows us to apply the Woodbury matrix identity:

$$(M_A + UV^\top)^{-1} = M_A^{-1} - M_A^{-1}U(I_2 + V^\top M_A^{-1}U)^{-1}V^\top M_A^{-1}.$$

Because $V^\top M_A^{-1}U$ is only a 2×2 matrix, evaluating this update is extremely fast. We can plug this inverted matrix directly back into Eq. (11), reducing the computational complexity of evaluating a neighbor to just $\mathcal{O}(|A|^2)$ instead of $\mathcal{O}(|A|^3)$.

Once the optimal batch $A^\star$ is found, we can proceed in two different ways: (i) **MYO**: We evaluate the target model on all items in $A^\star$ simultaneously. Batch evaluation allows us to maximize throughput when multiple evaluations can be processed in parallel; or (ii) **NON-MYO**: We select a single item from it at random. Because this item is selected based on an optimal future horizon rather than just the immediate next step, this renders the active selection non-myopic.

Figure 3 presents the RMSE LOG, RMSE GAIN, and COST-ACCURACY TRADE-OFF for GPQA with binary (left two columns) and continuous (right two columns) scores. We evaluate the selection strategies outlined above using batch sizes of $|A| = 2$ and $|A| = 5$. We observe that the overall performance of these batch approaches is similar to that of the single-item selection strategy.

Table 2 compares the EVALUATION-WEIGHTED INTEGRATED RMSE achieved by BAYESAME with single-candidate selection, random selection, batch selection with $|A| = 2$ using both myopic (MYO) and non-myopic (NON-MYO) criteria, and BAYES-AIPW across the different benchmarks. Consistent with the results of Figure 3, BAYESAME with batch size $|A| = 2$ exhibits performance similar to that of myopic single-candidate selection under both the myopic and non-myopic variants.

B.2 METADATA-WEIGHTED PRIOR DISTRIBUTION

When rich metadata (such as model size, training data, or hyperparameter configurations) is available for both the reference and target models, we can formulate a weighted prior distribution. Specifically, we define a weight vector $w = (w_1, \ldots, w_{K_r})$, where $w_k \geq 0$ captures the prior positive correlation between reference model k and the target model, with $\sum_{k=1}^{K_r} w_k = 1$. We then consider the prior distribution $p(\theta) = \mathcal{N}(\mu^\theta, \Sigma^\theta)$ with

$$\mu_b^\theta = \phi_b^\top w, \qquad \Sigma_{b,b'}^\theta = \alpha \exp\left(-\beta(\phi_b - \phi_{b'})^\top W(\phi_b - \phi_{b'})\right),$$

where $W = \mathrm{diag}(w)$. In practice, public leaderboards rarely provide sufficiently detailed metadata to calibrate these weights reliably, so we assign uniform weights across the reference models $w_k = 1/K_r$.

B.3 Logit-Normal Prior Distribution

While achieving strong empirical performance, a Gaussian prior distribution on θ is not the best choice for binary and bounded continuous scores from a modeling viewpoint, as it assigns positive probability mass outside the true support. Below, we describe more appropriate alternative formulations that we considered.

Case $s_i \in (0, 1)$ We introduce a logit-normal model that respects the support of scores. Specifically, we map both the latent abilities and the observed scores to the unconstrained real line using the logit transformation. We define the latent variable vector $f = (f_1, \ldots, f_B)$, where $f_b = \mathrm{logit}(\theta_b) = \ln\left(\frac{\theta_b}{1-\theta_b}\right)$, and assign a Gaussian prior distribution on it. The scores are similarly transformed, and we assume a Gaussian conditional distribution $\mathrm{logit}(S_i) \mid f_b \sim \mathcal{N}(f_b, N_b\sigma^2)$ for an item i belonging to bucket b. The posterior distribution of $f = (f_1, \ldots, f_B)$ conditioned on $s_{\mathcal{C}}$ is given by:

$$p(f \mid s_{\mathcal{C}}) \propto p(f) \prod_{i \in \mathcal{C}} p(\mathrm{logit}(s_i) \mid f).$$

Under this specification, $p(f \mid s_{\mathcal{C}})$ is itself Gaussian and inference over f remains fully tractable.

Case $s_i \in \{0, 1\}$ Similarly to above, consider the latent variable vector $f = (f_1, \ldots, f_B)$, where $f_b = \mathrm{logit}(\theta_b)$, and assign a Gaussian prior distribution on it. In the case of binary responses, a more appropriate model would use a Bernoulli likelihood of the form

$$p(s_i \mid f) = \sigma(f_b)^{s_i}(1 - \sigma(f_b))^{1-s_i},$$

for an item i belonging to bucket b, where σ denotes the logistic function. Then, the posterior distribution of $f = (f_1, \ldots, f_B)$ conditioned on $s_{\mathcal{C}}$ is given by:

$$p(f \mid s_{\mathcal{C}}) \propto p(f) \prod_{i \in \mathcal{C}} p(s_i \mid f).$$

In this case, we can resort to variational inference or a Laplace approximation (Rasmussen & Williams, 2006) to compute the posterior.

However, we are interested in the posterior of $\theta = \sigma(f) = (1 + e^{-f})^{-1}$, conditioned on $S_{\mathcal{C}} = s_{\mathcal{C}}$. The posterior mean can be computed as:

$$\mathbb{E}[\theta_b \mid s_{\mathcal{C}}] = \int \sigma(f_b) p(f \mid s_{\mathcal{C}}) \, \mathrm{d}f,$$

which does not admit a closed-form solution because of the nonlinearity of σ. We consider three strategies for approximating this integral.

- **Monte Carlo estimation.** The main drawback is that obtaining accurate estimates for all N items requires a large number of samples, which is prohibitive inside the sequential active learning loop.
- **Other numerical integration methods.** We can approximate $\sigma(f_b)$ in the integral with a rescaled probit function that has the same slope at the origin (MacKay, 1992), obtaining:

$$\mathbb{E}[\theta_b \mid s_{\mathcal{C}}] \approx \sigma\left(\frac{\mathbb{E}[f_b \mid s_{\mathcal{C}}]}{\sqrt{1 + \pi \, \mathrm{Var}(f_b \mid s_{\mathcal{C}})/8}}\right).$$

- **Linear approximation.** Following Osborne et al. (2012), we linearize the logistic term around a reference point $f_{b,0}$, that is,

$$\sigma(f_b) \approx \sigma(f_{b,0}) + \sigma(f_{b,0})(1 - \sigma(f_{b,0}))(f_b - f_{b,0})$$

  Under this approximation, the posterior mean of θ admits a closed-form expression. To select the reference point, we introduce an auxiliary multivariate Gaussian model in the original space and set $\sigma(f_{b,0})$ to be its posterior mean given $s_{\mathcal{C}}$, i.e., $\sigma(f_{b,0}) = \mathbb{E}[\theta_b \mid s_{\mathcal{C}}]$, clipping the value to an interior interval $[\eta, 1 - \eta]$, $0 < \eta < 1/2$, before taking the logit. Although this auxiliary Gaussian model is not appropriate as it does not have the correct range, it is used solely to inform the choice of $f_{b,0}$.

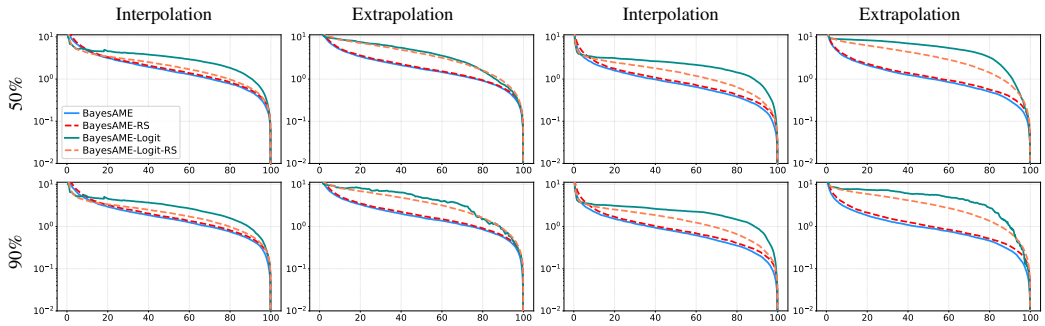

Figure 4: **Single-target Setting. GPQA** with binary scores (left two columns) and continuous scores (right two columns). RMSE LOG across 50% and 90% of reference models.

While these alternative formulations offer a more principled treatment of the score support, they require approximate inference at every step and introduce substantial computational overhead. In practice, we found that these approximations resulted in worse performance. For example, Figure 4 compares BAYESAME and BAYESAME-RS with their corresponding versions using logit-normal models (BAYESAME-LOGIT and BAYESAME-LOGIT-RS) on GPQA for both binary and continuous scores, using the linear approximation described above to derive the posterior for θ. For both settings and regardless of the selection strategy, BAYESAME and BAYESAME-RS outperform the logit-normal formulation.

C EXPERIMENTAL DETAILS AND ADDITIONAL RESULTS

C.1 BENCHMARKS

We evaluate our approach across six standard benchmarks collected from three primary sources: the Open LLM Leaderboard (Fourrier et al., 2024), evaluations compiled by Zhang et al. (2025), and the HELM Lite benchmark (Liang et al., 2023).

From the Open LLM Leaderboard, we extracted scores for GPQA (Rein et al., 2024), MMLU-Pro (Wang et al., 2024; Hendrycks et al., 2021), BBH (Suzgun et al., 2023), and MuSR (Sprague et al., 2024). Beyond standard binary scores, we leveraged the leaderboard's raw log-likelihoods to obtain continuous scores via a softmax transformation. These continuous scores allow us to assess the impact of exploiting richer information and mitigate the issue of identical reference model scores. To ensure a representative distribution of model capabilities, we partitioned the available models into five quantiles based on accuracy. We then sampled evenly by selecting the most popular models from each quantile, resulting in an initial set of 300 models. After extracting the data for these selections, we filtered out any records containing NaN values caused by incomplete leaderboard entries, formatting discrepancies, or evaluation timeouts. This curation process yielded the following final model counts per benchmark: GPQA: 102, MMLU-Pro: 165, BBH: 279, and MuSR: 288. For BBH, we only consider a subset of scenarios: temporal sequences, salient translation error detection, tracking shuffled objects (seven objects), geometric shapes, and reasoning about colored objects. For MMLU-Pro, we also restrict our evaluation to a subset of subjects, namely business and law, resulting in a total of 1890 items.

From Zhang et al. (2025), we extracted binary scores for IFEval (Zhou et al., 2023) across 448 models. Finally, from HELM Lite (up to version 1.13.0), we extracted the F1 scores for the Natural QA Openbook long-answer scenario (Kwiatkowski et al., 2019) across 44 models.

C.2 IMPLEMENTATION DETAILS

Interpolation and Extrapolation Reference Setup To investigate the interpolation and extrapolation regimes with granularity, we partition the available models into a pool of potential reference models and a fixed set of 10 target models, following an approach similar to Zhang et al. (2025). In the interpolation regime, the target models are selected uniformly at random, with all remaining

models assigned to the reference pool. In the extrapolation regime, we induce a distribution shift by designating the 10 highest-performing models as the target set, discarding the next 20% of models to create a performance gap, and using the remaining lower-performing models as potential reference models. To assess the impact of reference model quantity, we vary the size of the set of reference models by sampling 10%, 50%, or 90% of the reference pool uniformly at random.

BAYESAME For the single-target setting, we use Algorithm 1 with $\sigma^2 = 10^{-4}$ when $B \geq N - N_0$ and $\sigma^2 = 10^{-1}$ otherwise, $N_0 = 10$, $W = 30$. As described in Appendix A.4, optimization is performed using L-BFGS-B with $\alpha_0 = \beta_0 = 1$. Hyperparameters are optimized at every iteration for the first F observations, and every F-th iteration thereafter, with $F = 50$ (or $F = 100$ for benchmarks with more than 1500 items).

For the multi-target setting, we set $N_0 = 10$, and $W = 30$. Of the L covariances, half are squared exponential (Eq. (1)) and the rest are Matérn-3/2. As described in Appendix A.4, optimization is performed using AdamW, with α_l initialized to 1, β_l initialized to e^{-1}, each w_l^t is sampled from $\mathcal{N}(0, 0.01)$, and σ^2 initialized to 0.1 when $B < N - N_0$ and to 0.01 otherwise. Hyperparameters are re-optimized every $F = 5$ new observations, each time running 500 AdamW steps (reduced to 100 after 100 active learning iterations), with learning rate $\eta = 0.01$ for binary scores and $\eta = 0.001$ for continuous scores.

We set the margin $N_0 = 10$ to safely handle regimes where reference scores are nearly unique across items. Empirically, we identified a specific failure mode for active selection: when almost all items have unique combinations of reference scores (effectively forming singleton buckets), but one single bucket contains a significantly larger number of items, the active selection biases the posterior estimation and degrades performance. Enforcing $N_0 = 10$ prevents the algorithm from relying on active selection in this skewed regime, yielding stable results across all scenarios considered in this work. While fixing $N_0 = 10$ serves as a safe value, because the total number of buckets and the exact distribution of items across them are fully known to the algorithm, a more sophisticated future approach could explicitly incorporate this distributional information to set more suitable, benchmark-specific values. We conjecture that larger threshold values may be admissible when bucket sizes are strictly balanced, as the observed estimation bias appears to be driven by the presence of a single, severely unbalanced bucket. Note that when scores are continuous, reference vectors may be nearly identical without being exactly equal. We compute an approximate bucket count by greedily retaining items whose reference vector differs from all previously retained items by more than 0.01 in the ℓ_∞ norm, and default to random selection when this count falls below $N - N_0$. This criterion governs only the acquisition function, the model still uses exact buckets.

When selecting items in batches, we scale the window size W proportionally. Specifically, for a batch size $|A|$, we set $W = \lceil 30/|A| \rceil$.

C.3 BASELINES

Since all existing approaches that leverage reference models assume a predefined coreset size, we adapt the most effective methods to operate in the same sequential manner as Algorithm 1. Specifically, $\mathcal{C}$ is iteratively augmented until $\max_{l \in [j-W+1,j]}(\hat{R}_l) - \min_{l \in [j-W+1,j]}(\hat{R}_l) \leq \epsilon_1$ and $2z_{0.975}\sqrt{\mathrm{Var}_R} \leq \epsilon_2$ where $\hat{R}$ and Var_R are defined below.

1. **BAYESAME-RS**: A simpler version of BAYESAME that selects items at random.
2. **SEQ-APW**: An extension of the anchor points weighted method from Vivek et al. (2024), in which $\mathcal{C}$ is augmented by choosing items that maximize space covering, as measured by correlations between the reference models' performances. Benchmark items are partitioned into clusters by assigning each item to its nearest anchor in $\mathcal{C}$. $\hat{R} = \sum_{i \in \mathcal{C}} w_i s_i$, where w_i is a weight proportional to the cluster size, while Var_R is given by the following estimator of $\mathrm{Var}(R)$:

$$\mathrm{Var}_R = s^2 \left(\sum_{i \in \mathcal{C}} w_i^2 \right) \frac{N - n_{\text{eff}}}{N}, \quad n_{\text{eff}} = \frac{\left(\sum_{i \in \mathcal{C}} w_i \right)^2}{\sum_{i \in \mathcal{C}} w_i^2}, \tag{13}$$

where s denotes the sample standard deviation. By construction, $\hat{R}$ recovers the true performance $R^\star$ once the entire benchmark has been evaluated. Moreover, Var_R captures the sampling

variance arising from the randomness in the coreset selection. Due to the finite-population correction factor, this variance decreases as the coreset grows and converges to zero when the full benchmark is evaluated.

3. **RS-MEAN**: The simplest baseline that augments $\mathcal{C}$ by selecting items at random and estimates the full benchmark performance as the empirical average performance on $\mathcal{C}$. Var_R is given by Eq. (13) using $w_i = 1/n$. The same properties regarding $\hat{R}$ and Var_R discussed for SEQ-APW apply.
4. **BAYES-AIPW**: A Bayesian ridge regression extension of AIPW (Zhang et al., 2025), obtained by placing a Gaussian prior distribution on the coefficients β of the model $s_i = \phi_i^\top \beta + \varepsilon_i$, where $\phi_i = (s_i^{r_1}, \ldots, s_i^{r_{K_r}})$, and $\varepsilon_i \sim \mathcal{N}(0, \alpha^{-1})$ with $\alpha \sim \mathrm{Gamma}(\alpha_1, \alpha_2)$. By conditioning on $\{(\phi_i, s_i)\}_{i \in \mathcal{C}}$, we obtain a posterior distribution over β with mean $\mu^\beta_{|\mathcal{C}}$ and covariance $\Sigma^\beta_{|\mathcal{C}}$. As $\hat{R}$, we use the AIPW estimator:

$$\hat{R} = \tfrac{1}{n}\textstyle\sum_{i \in \mathcal{C}} s_i + \frac{1}{1+\frac{n}{N-n}} \left(\frac{1}{N-n} \sum_{i \in E \setminus \mathcal{C}} \mu_{i|\mathcal{C}} - \frac{1}{n} \sum_{i \in \mathcal{C}} \mu_{i|\mathcal{C}} \right),$$

where $\mu_{i|\mathcal{C}} = \phi_i^\top \mu^\beta_{|\mathcal{C}}$. Note that $\hat{R}$ recovers the true performance $R^\star$ when all items have been evaluated. Posterior uncertainty is obtained by propagating the posterior distribution of β through the AIPW correction, yielding:

$$\mathrm{Var}_R = \frac{(\bar{\phi}_{E\setminus\mathcal{C}} - \bar{\phi}_{\mathcal{C}})^\top \Sigma^\beta_{|\mathcal{C}} (\bar{\phi}_{E\setminus\mathcal{C}} - \bar{\phi}_{\mathcal{C}}) + \hat{\alpha}^{-1} \left(\frac{1}{n} + \frac{1}{N-n}\right)}{\left(1 + \frac{n}{N-n}\right)^2},$$

where $\bar{\phi}_{\mathcal{C}} = \frac{1}{n}\sum_{i \in \mathcal{C}} \phi_i$, $\bar{\phi}_{E\setminus\mathcal{C}} = \frac{1}{N-n}\sum_{i \in E\setminus\mathcal{C}} \phi_i$, and $\hat{\alpha}$ is the posterior mean of α. $\mathcal{C}$ is augmented by selecting items at random. Figure 5 in Appendix C.4 shows that this Bayesian extension matches or exceeds the performance of the original AIPW method.

5. **BAYES-RS-LEARN**: A Bayesian ridge regression extension of RANDOM-SAMPLING-LEARN (Zhang et al., 2025), obtained by placing a Gaussian prior on the coefficients β of the model $R = s_{\mathcal{C}}^\top \beta + \varepsilon$, where $\varepsilon \sim \mathcal{N}(0, \alpha^{-1})$, $\alpha \sim \mathrm{Gamma}(\alpha_1, \alpha_2)$. We perform inference on β by conditioning on $\{(s_{\mathcal{C}}^{r_k}, R^\star_{r_k})\}_{k=1}^{K_r}$. This yields a posterior distribution over β with mean $\mu^\beta_{|\mathcal{C}}$ and covariance $\Sigma^\beta_{|\mathcal{C}}$. We then obtain $\hat{R}$ and Var_R as:

$$\hat{R} = s_{\mathcal{C}}^\top \mu^\beta_{|\mathcal{C}}, \quad \mathrm{Var}_R = s_{\mathcal{C}}^\top \Sigma^\beta_{|\mathcal{C}} s_{\mathcal{C}} + \hat{\alpha}^{-1},$$

where $\hat{\alpha}^{-1}$ is the posterior mean of α^{-1}.

6. **PROEVAL**: The method in Huang et al. (2026) for the setting with no prior performance data for the target model (called New Model setting with Score Features in the original paper), without abstention rule. The method defines the target model's performance as $R = \frac{1}{N}\sum_{i=1}^N \theta_i$, where $\theta = (\theta_1, \ldots, \theta_N)$ is a Gaussian random vector whose prior covariance matrix is given by the empirical covariance of the reference model scores. $\hat{R} = \mathbb{E}[R \,|\, s_{\mathcal{C}}]$ and $\mathrm{Var}_R = \mathrm{Var}(R \,|\, s_{\mathcal{C}})$. Because R is defined as the average of the latent abilities, the estimator $\hat{R}$ can deviate significantly from the true performance $R^\star$ even for large coreset sizes, as discussed in Section 4. The confidence interval induced by Var_R can be interpreted as a Bayesian credible interval.
7. **IRT**: Two variants of IRT where $\mathcal{C}$ is selected randomly, depending on the score type: one for binary scores based on the 2-PL model (Polo et al., 2024) (P-IRT) and one for continuous scores based on the Beta distribution, specifically the extension of Noel & Dauvier (2007) provided in Chen et al. (2019). The item parameters are learned using the reference models' scores, and the scores $s_{\mathcal{C}}$ are used to fit the target model-specific parameters. As the performance estimator, we deploy Eq. (5) with $w = 1/N$. Because we fit the model parameters separately for each benchmark, treating them as scalars rather than vector-valued as in Polo et al. (2024), applying the generalized variant (GP-IRT) does not improve performance. We fit the parameters using MCMC. This provides samples from the posterior distribution of the parameters, allowing us to directly obtain the posterior distribution of R and its variance, which we use as $\hat{R}$ and Var_R. Due to the computational cost of MCMC, we update the parameters only every 50 evaluations (resulting in piecewise-constant curves in the plots).

For BAYES-AIPW, BAYES-RS-LEARN, and SEQ-APW, we adapt the implementations of AIPW, RANDOM-SAMPLING-LEARN, and APW, respectively, provided by Zhang et al. (2025) in `https://github.com/socialfoundations/benchmark-prediction`. In BAYES-AIPW and BAYES-RS-LEARN, we replace the linear ridge regression with its Bayesian counterpart as implemented in `scikit-learn` library using the default setting (Pedregosa et al., 2011). For PROEVAL, we use the implementation provided by Huang et al. (2026) in `https://github.com/google-deepmind/proeval`. For IRT, we implement the binary and continuous approaches based on the methodology outlined in Polo et al. (2024) and Chen et al. (2019), respectively.

C.4 ADDITIONAL RESULTS SINGLE TARGET SETTING

Figure 5 shows that our Bayesian extension of AIPW matches or outperforms the original method of Zhang et al. (2025).

We present extended results for all benchmarks considered. To facilitate navigation of the plots, Table 3 summarizes the figures corresponding to each benchmark and evaluation metric.

Table 3: Figure index for the extended experimental results across all benchmarks.

PLOT TYPE	GPQA	MMLU-PRO	BBH	MUSR	IFEVAL	NATURAL QA OPENBOOK
RMSE LOG, RMSE GAIN & Cost-Accuracy Tradeoff	6	7	8	9	10	10
SPEARMAN CORRELATION	11	–	11	–	–	–

C.5 MULTI-TARGET SETTING

Figure 12 shows the results comparing BAYESAME MULTI-TARGET with BAYESAME SINGLE-TARGET on GPQA using both binary and continuous scores.

Figure 13 reports results for the alternative selection strategy introduced in Section 3.2, which selects items based on the average marginalized information gain across target models.

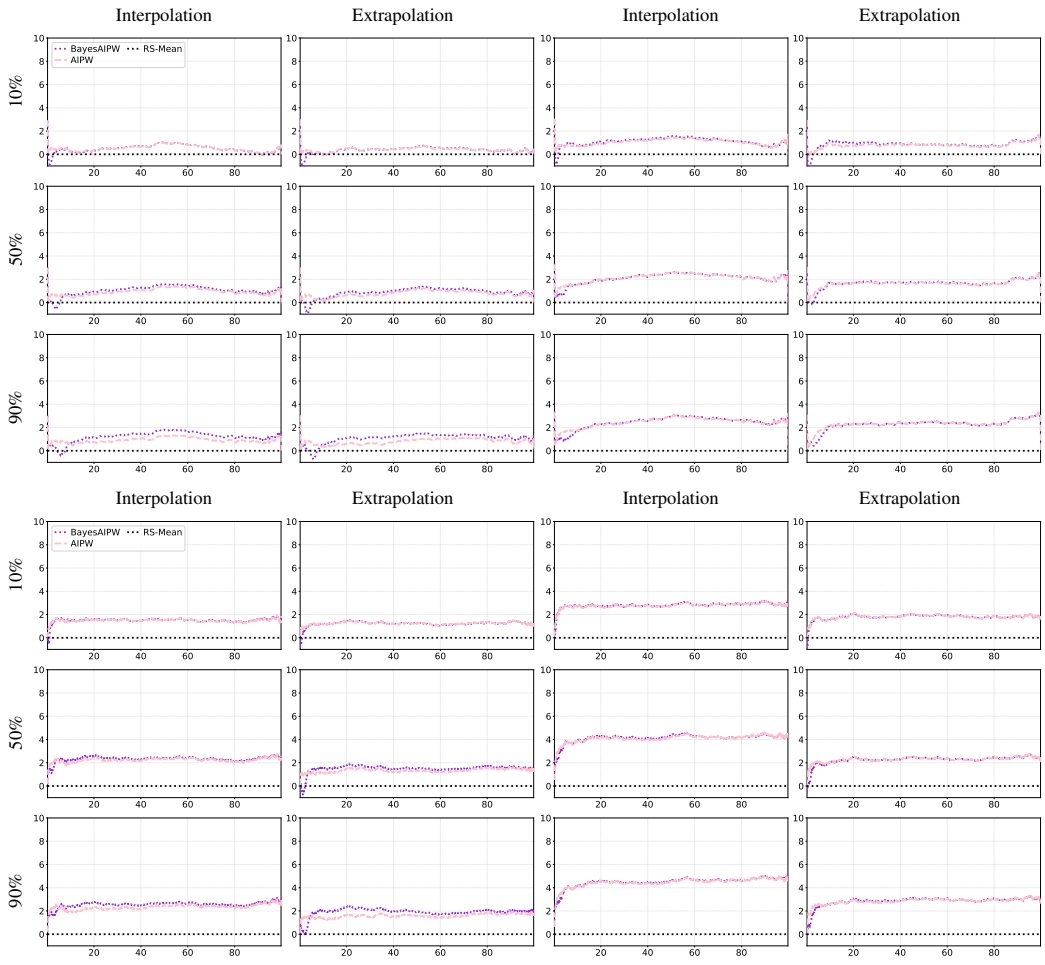

Figure 5: **Single-target Setting. GPQA** (top three rows) and **MMLU-Pro** (bottom three rows) with binary scores (left two columns) and continuous scores (right two columns). RMSE GAIN for the AIPW ablation study across varying proportions of reference models (10%, 50%, and 90%) ablation for AIPW.

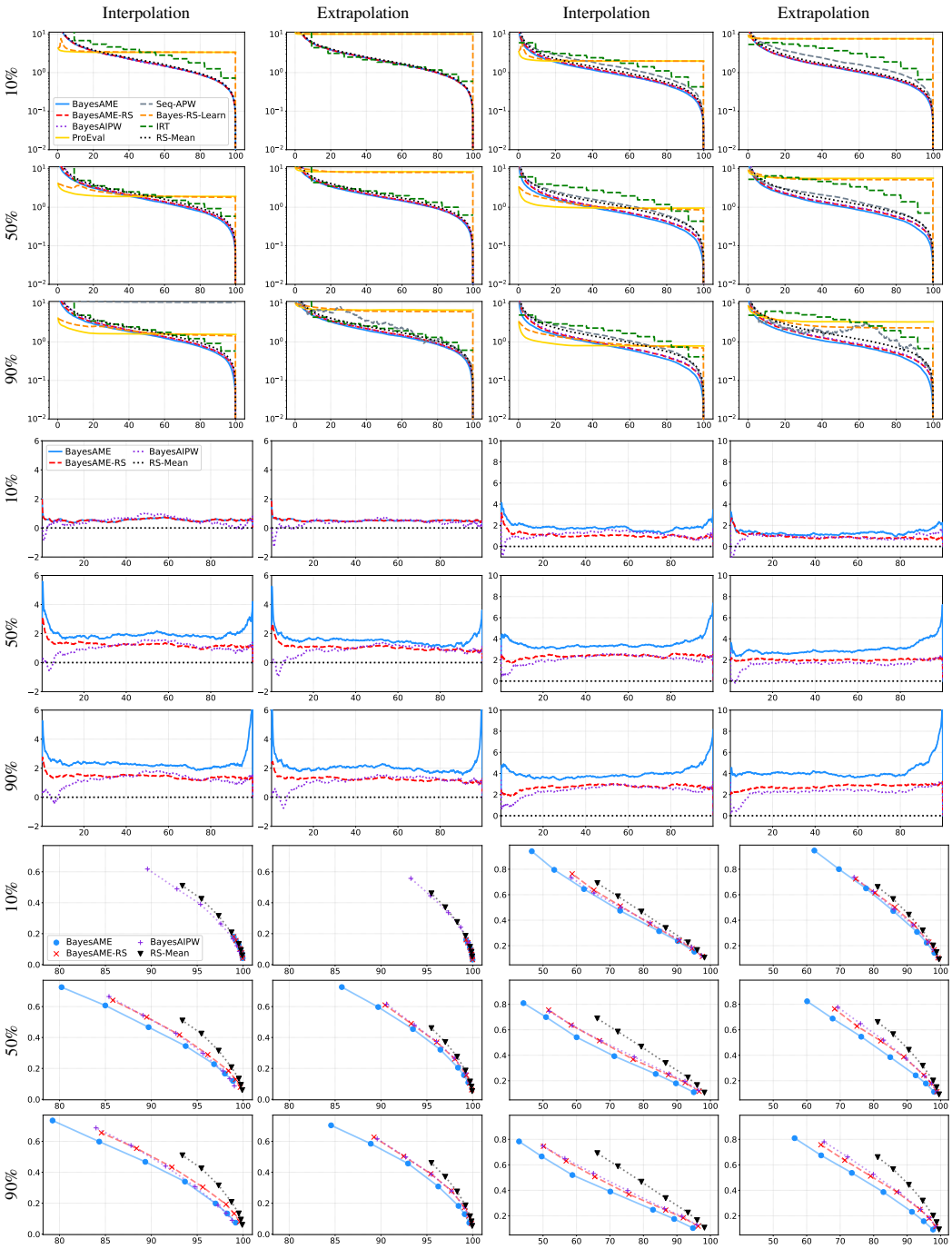

Figure 6: **Single-target Setting. GPQA** with binary scores (left two columns) and continuous scores (right two columns). RMSE LOG (top three rows), RMSE GAIN (middle three rows), and COST-ACCURACY TRADE-OFF (bottom three rows) across varying proportions of reference models.

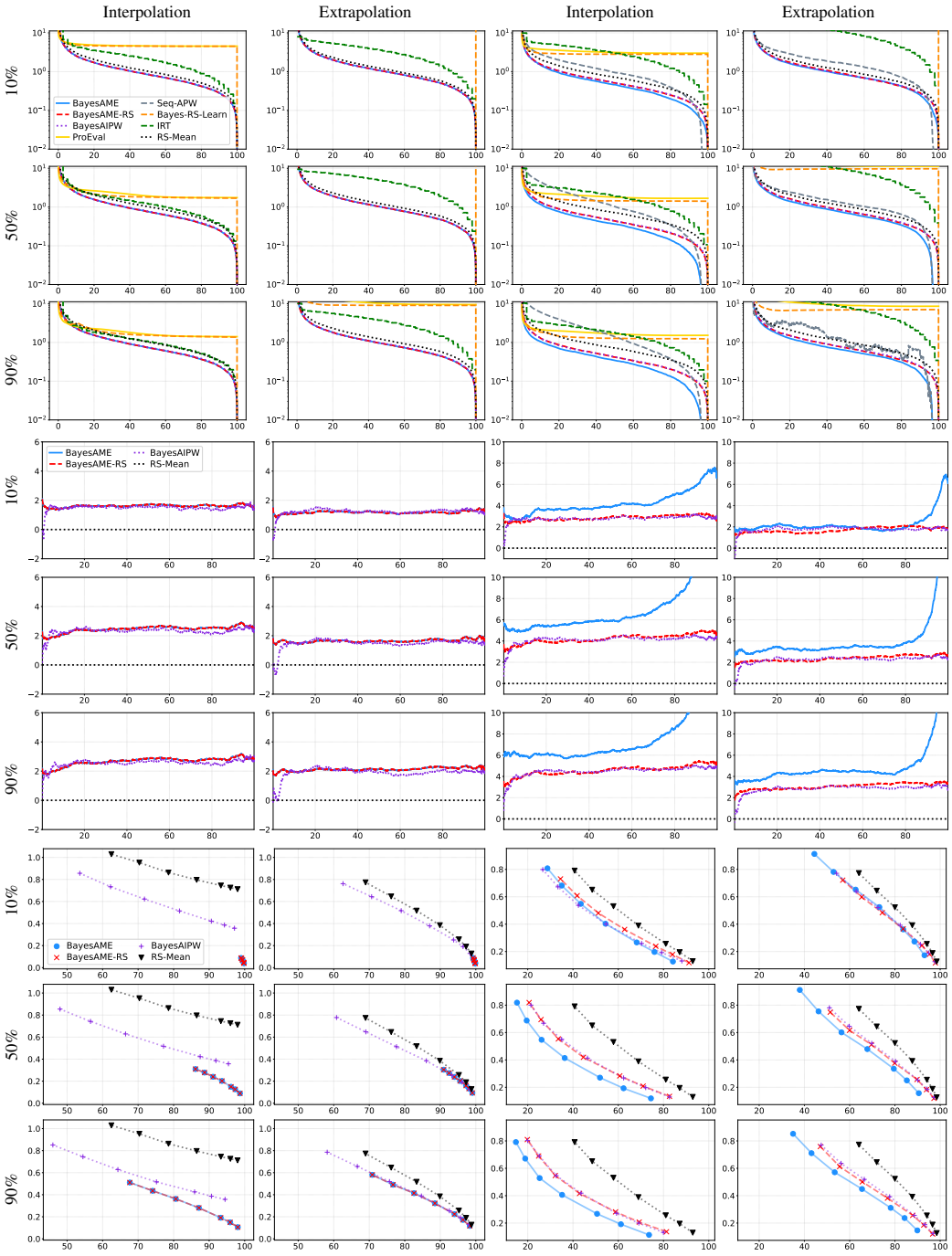

Figure 7: **Single-target Setting. MMLU-Pro** with binary scores (left two columns) and continuous scores (right two columns). RMSE LOG (top three rows), RMSE GAIN (middle three rows), and COST-ACCURACY TRADE-OFF (bottom three rows) across varying proportions of reference models.

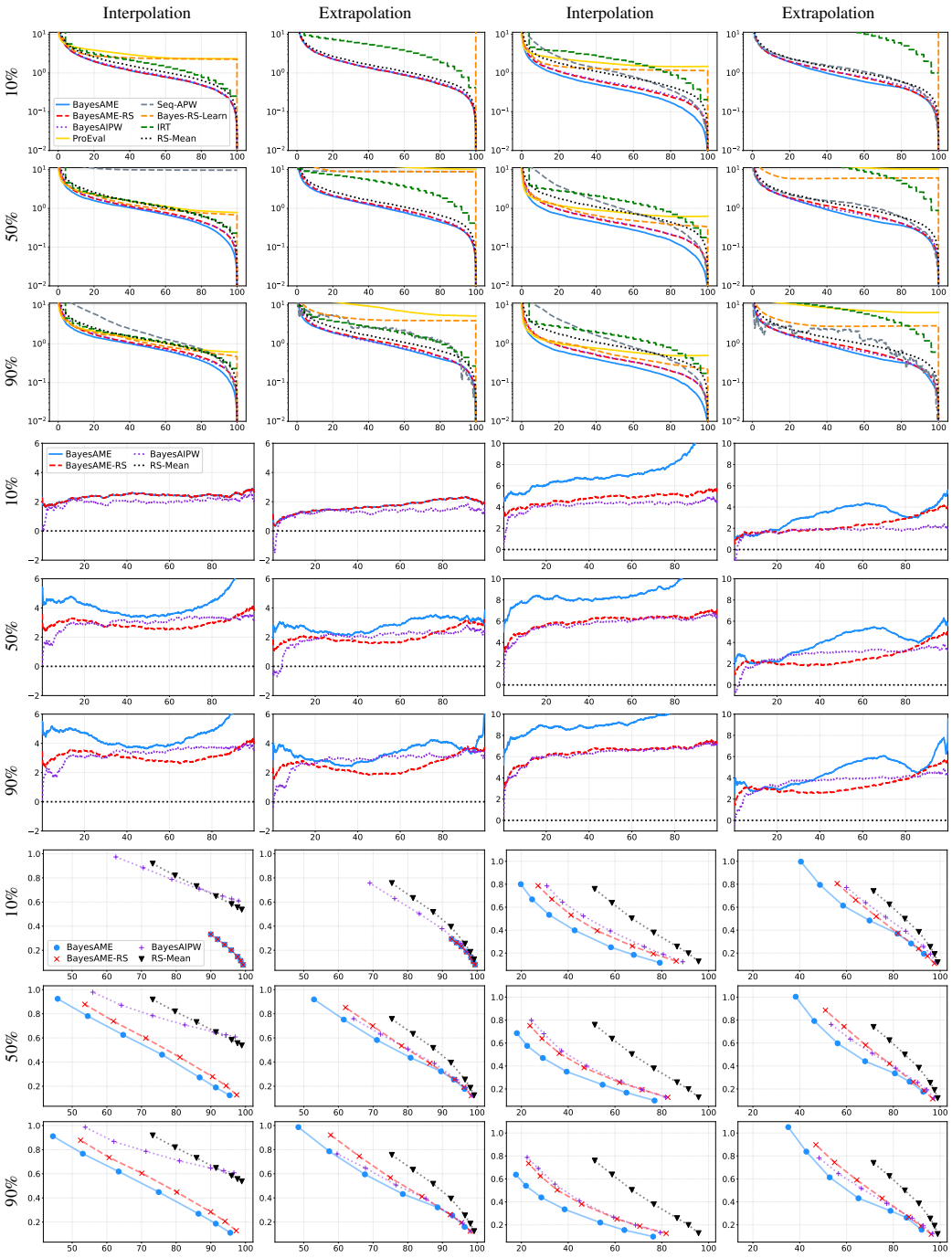

Figure 8: **Single-target Setting. BBH** with binary scores (left two columns) and continuous scores (right two columns). RMSE LOG (top three rows), RMSE GAIN (middle three rows), and COST-ACCURACY TRADE-OFF (bottom three rows) across varying proportions of reference models.

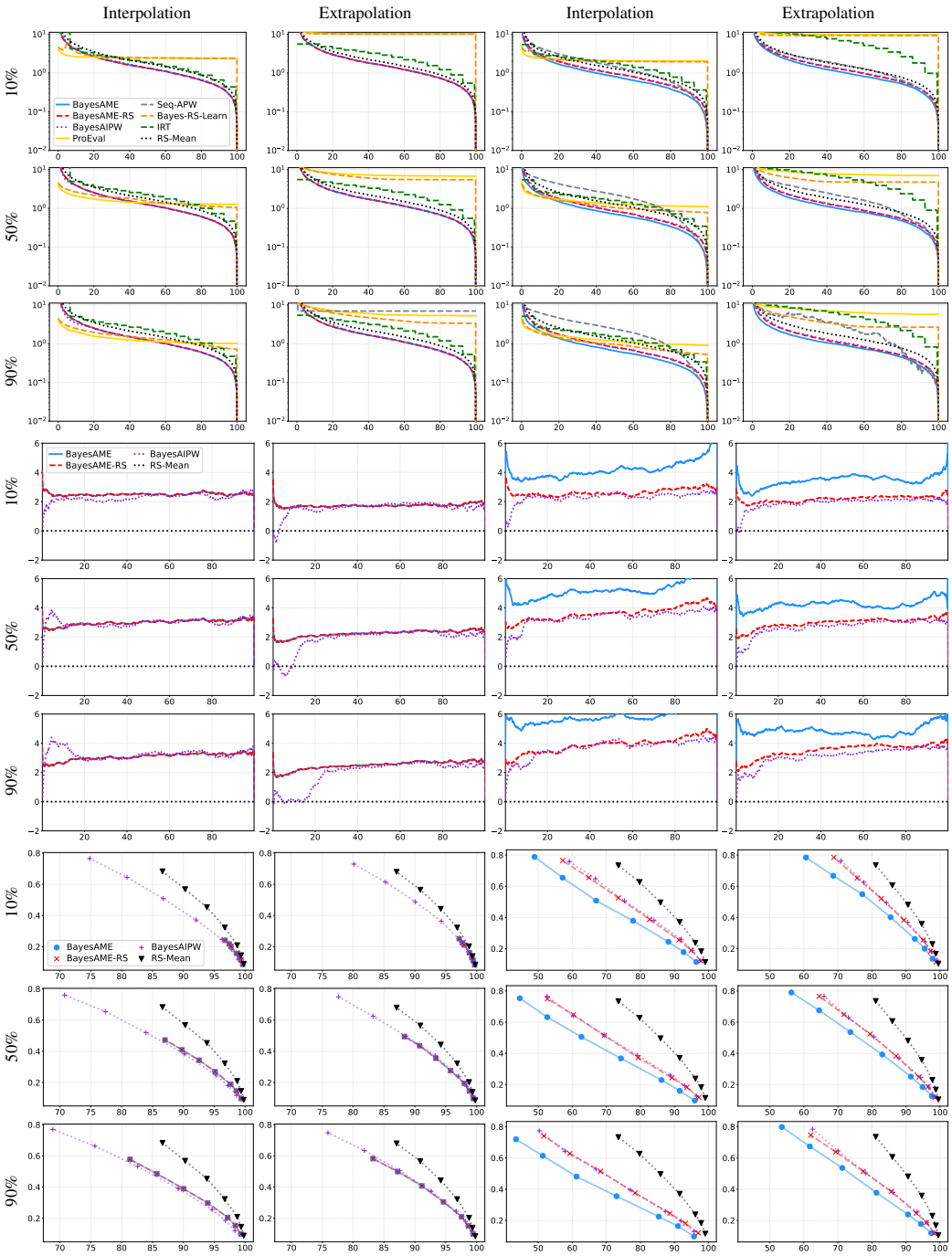

Figure 9: **Single-target Setting. MuSR** with binary scores (left two columns) and continuous scores (right two columns). RMSE LOG (top three rows), RMSE GAIN (middle three rows), and COST-ACCURACY TRADE-OFF (bottom three rows) across varying proportions of reference models.

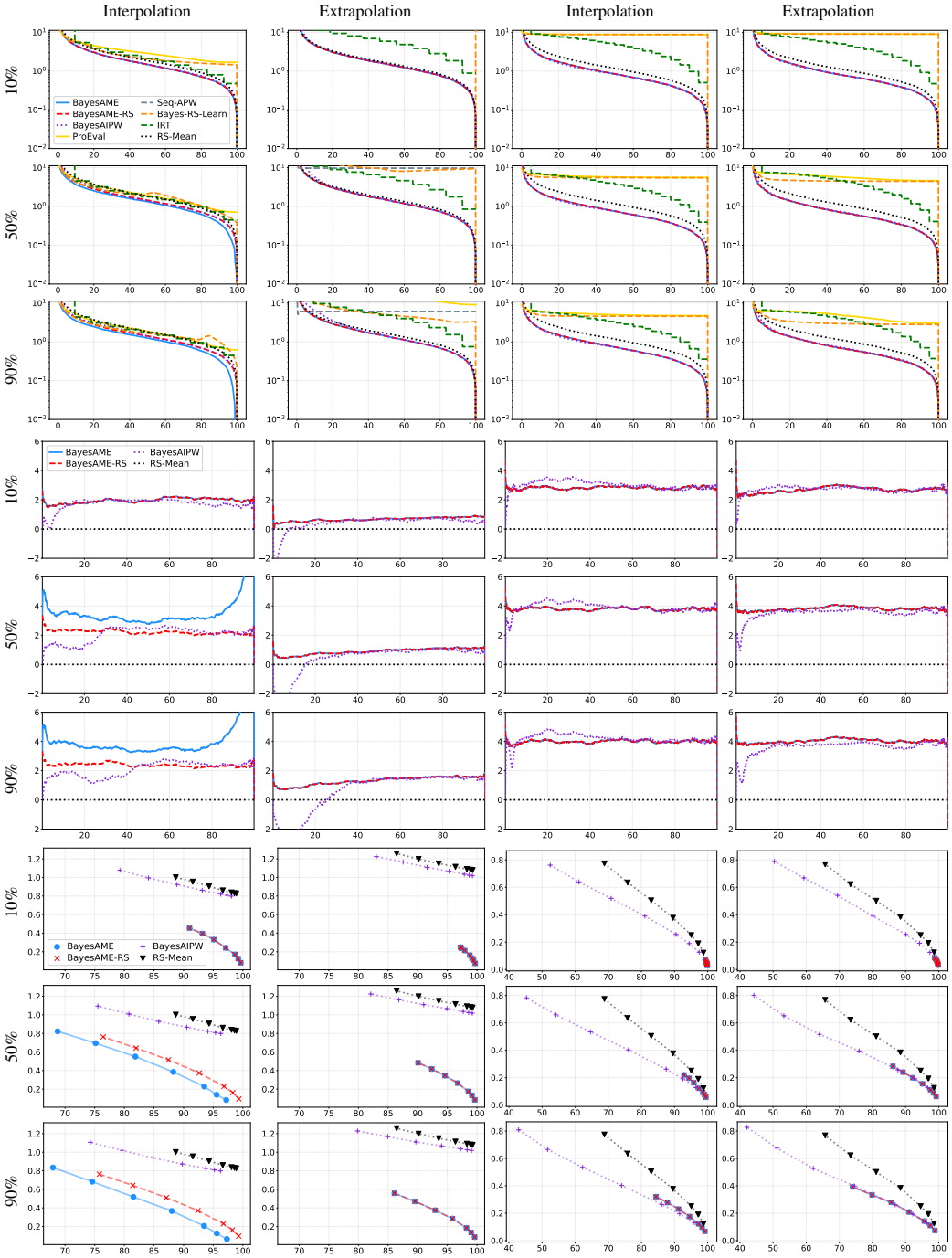

Figure 10: **Single-target Setting. IFEval** with binary scores (left two columns) and **Natural QA Openbook** with continuous scores (right two columns). RMSE LOG (top three rows), RMSE GAIN (middle three rows), and COST-ACCURACY TRADE-OFF (bottom three rows) across varying proportions of reference models.

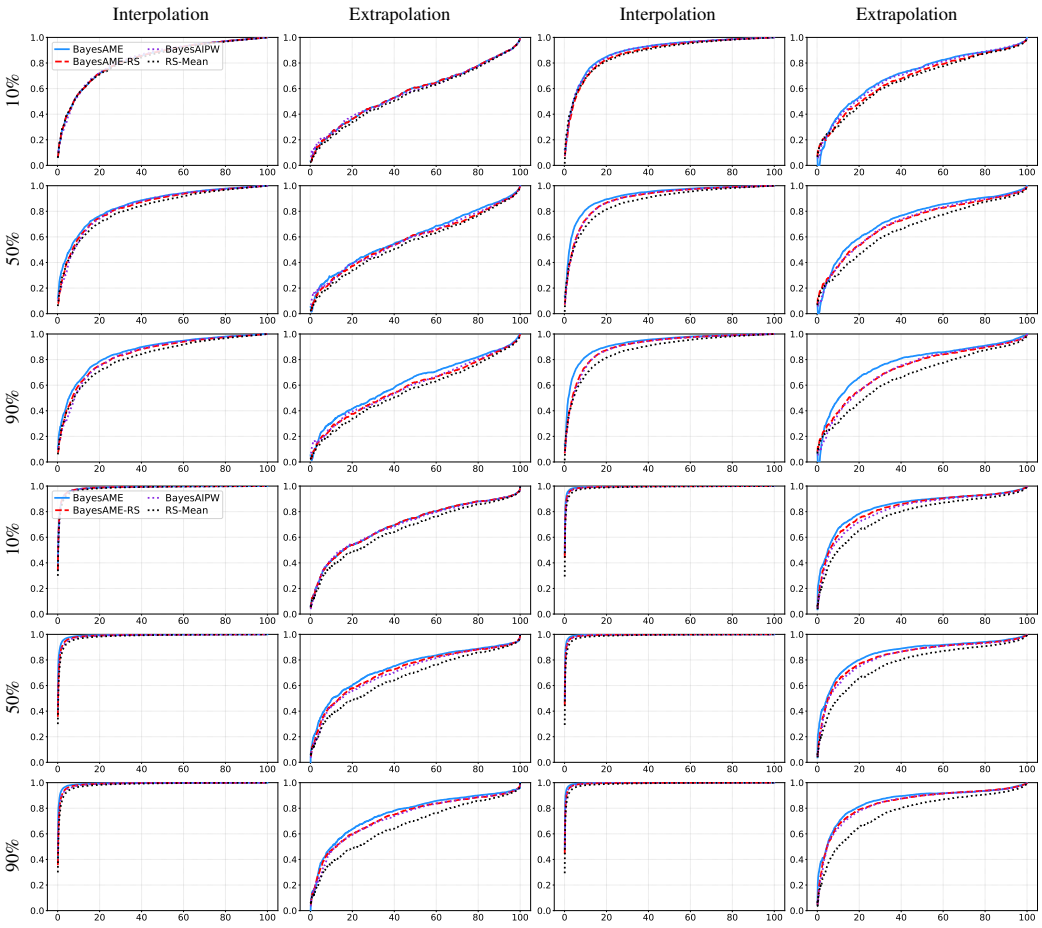

Figure 11: **Single-target Setting. GPQA** (top three rows) and **BBH** (bottom three rows) with binary scores (left two columns) and continuous scores (right two columns). Spearman's rank correlation across varying proportions of reference models.

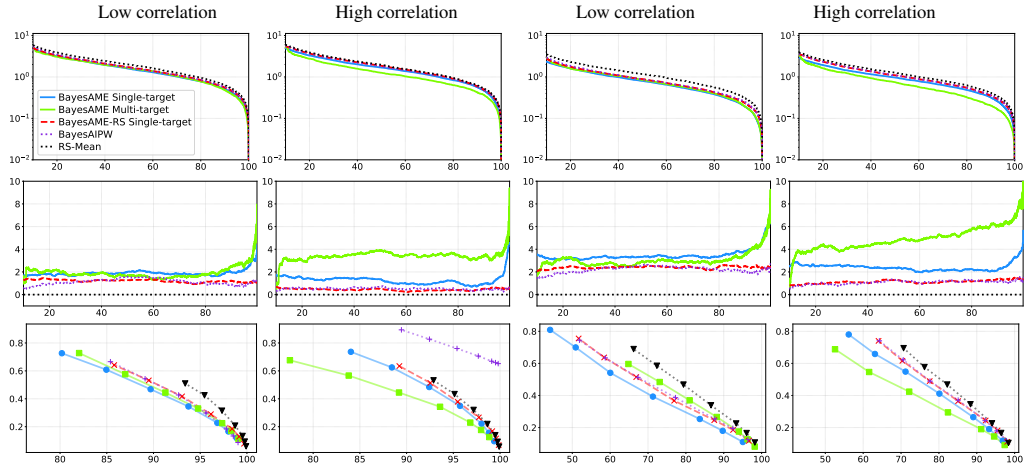

Figure 12: **Multiple-target Setting. GPQA** with binary scores (left two columns) and with continuous scores (right two columns). The rows represent RMSE LOG, RMSE GAIN, and COST-ACCURACY TRADE-OFF, respectively. We compare the multitarget method against methods that do not model correlations across targets.

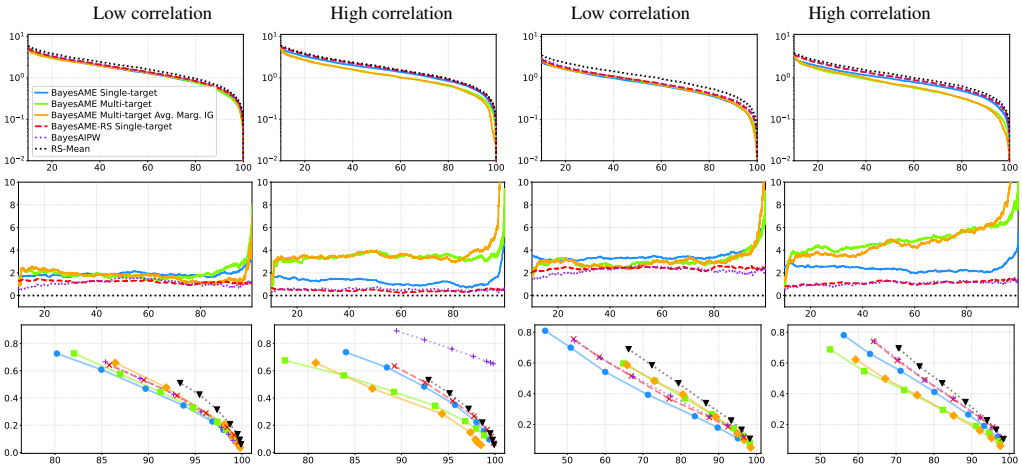

Figure 13: **Multiple-target Setting. GPQA** with binary scores (left two columns) and with continuous scores (right two columns). The rows represent RMSE LOG, RMSE GAIN, and COST-ACCURACY TRADE-OFF, respectively. We compare the different selection strategies in the multi-target setting.